

Network Working Group
Request for Comments: 1493
Obsoletes: 1286

E. Decker
cisco Systems, Inc.
P. Langille
Digital Equipment Corporation
A. Rijsinghani
Digital Equipment Corporation
K. McCloghrie
Hughes LAN Systems, Inc.
July 1993

Definitions of Managed Objects for Bridges

Status of this Memo

This RFC specifies an IAB standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP based internets. In particular it defines objects for managing MAC bridges based on the IEEE 802.1D-1990 standard between Local Area Network (LAN) segments. Provisions are made for support of transparent bridging. Provisions are also made so that these objects apply to bridges connected by subnetworks other than LAN segments.

Table of Contents

1. The Network Management Framework	2
2. Objects	2
2.1 Format of Definitions	3
3. Overview	3
3.1 Structure of MIB	3
3.1.1 The dot1dBase Group	6
3.1.2 The dot1dStp Group	6
3.1.3 The dot1dSr Group	6
3.1.4 The dot1dTp Group	6
3.1.5 The dot1dStatic Group	6
3.2 Relationship to Other MIBs	6
3.2.1 Relationship to the 'system' group	6
3.2.2 Relationship to the 'interfaces' group	7

3.3 Textual Conventions	8
4. Changes from RFC 1286	8
5. Definitions	9
5.1 Groups in the Bridge MIB	11
5.2 The dot1dBase Group Definitions	11
5.3 The dot1dStp Group Definitions	14
5.4 The dot1dTp Group Definitions	22
5.5 The dot1dStatic Group Definitions	28
5.6 Traps for use by Bridges	31
6. Acknowledgments	31
7. References	33
8. Security Considerations	33
9. Authors' Addresses	34

1. The Network Management Framework

The Internet-standard Network Management Framework consists of three components. They are:

STD16/RFC 1155 which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management.
STD16/RFC 1212 defines a more concise description mechanism, which is wholly consistent with the SMI.

RFC 1156 which defines MIB-I, the core set of managed objects for the Internet suite of protocols. STD17/RFC 1213, defines MIB-II, an evolution of MIB-I based on implementation experience and new operational requirements.

STD15/RFC 1157 which defines the SNMP, the protocol used for network access to managed objects.

The Framework permits new objects to be defined for the purpose of experimentation and evaluation.

2. Objects

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the subset of Abstract Syntax Notation One (ASN.1) [7] defined in the SMI. In particular, each object is named by an OBJECT IDENTIFIER, an administratively assigned name, which specifies an object type. The object type together with an object instance serves to uniquely identify a specific instantiation of the object. For human convenience, we often use a textual string, termed the descriptor, to also refer to the object type.

2.1. Format of Definitions

Section 5 contains the specification of all object types contained in this MIB module. The object types are defined using the conventions defined in the SMI, as amended by the extensions specified in [9,10].

3. Overview

A common device present in many networks is the Bridge. This device is used to connect Local Area Network segments below the network layer.

There are two major modes defined for this bridging; transparent and source route. The transparent method of bridging is defined in the draft IEEE 802.1d specification [11]. This memo defines those objects needed for the management of a bridging entity operating in the transparent mode, as well as some objects applicable to all types of bridges.

To be consistent with IAB directives and good engineering practice, an explicit attempt was made to keep this MIB as simple as possible. This was accomplished by applying the following criteria to objects proposed for inclusion:

- (1) Start with a small set of essential objects and add only as further objects are needed.
- (2) Require objects be essential for either fault or configuration management.
- (3) Consider evidence of current use and/or utility.
- (4) Limit the total of objects.
- (5) Exclude objects which are simply derivable from others in this or other MIBs.
- (6) Avoid causing critical sections to be heavily instrumented. The guideline that was followed is one counter per critical section per layer.

3.1. Structure of MIB

Objects in this MIB are arranged into groups. Each group is organized as a set of related objects. The overall structure and assignment of objects to their groups is shown below. Where appropriate the corresponding IEEE 802.1d [11] management object name is also included.

Bridge MIB Name	IEEE 802.1d Name
dot1dBridge	
dot1dBase	
BridgeAddress	Bridge.BridgeAddress
NumPorts	Bridge.NumberOfPorts
Type	
PortTable	
Port	BridgePort.PortNumber
IfIndex	
Circuit	
DelayExceededDiscards	.DiscardTransitDelay
MtuExceededDiscards	.DiscardOnError
dot1dStp	
ProtocolSpecification	
Priority	SpanningTreeProtocol
	.BridgePriority
TimeSinceTopologyChange	.TimeSinceTopologyChange
TopChanges	.TopologyChangeCount
DesignatedRoot	.DesignatedRoot
RootCost	.RootCost
RootPort	.RootPort
MaxAge	.MaxAge
HelloTime	.HelloTime
HoldTime	.HoldTime
ForwardDelay	.ForwardDelay
BridgeMaxAge	.BridgeMaxAge
BridgeHelloTime	.BridgeHelloTime
BridgeForwardDelay	.BridgeForwardDelay
PortTable	
Port	SpanningTreeProtocolPort
	.PortNumber
Priority	.PortPriority
State	.SpanningTreeState
Enable	
PathCost	.PortPathCost
DesignatedRoot	.DesignatedRoot
DesignatedCost	.DesignatedCost
DesignatedBridge	.DesignatedBridge
DesignatedPort	.DesignatedPort
ForwardTransitions	
dot1dTp	
LearnedEntryDiscards	BridgeFilter.DatabaseSize
	.NumDynamic,NumStatic
AgingTime	BridgeFilter.AgingTime
FdbTable	
Address	
Port	

```

        Status
    PortTable
        Port
        MaxInfo
        InFrames
        OutFrames
        InDiscards
    dot1dStatic
        StaticTable
            Address
            ReceivePort
            AllowedToGoTo
            Status
        BridgePort.FramesReceived
        .ForwardOutbound
        .DiscardInbound

```

The following IEEE 802.1d management objects have not been included in the Bridge MIB for the indicated reasons.

IEEE 802.1d Object	Disposition
Bridge.BridgeName	Same as sysDescr (MIB II)
Bridge.BridgeUpTime	Same as sysUpTime (MIB II)
Bridge.PortAddresses	Same as ifPhysAddress (MIB II)
BridgePort.PortName	Same as ifDescr (MIB II)
BridgePort.PortType	Same as ifType (MIB II)
BridgePort.RoutingType	Derivable from the implemented groups
SpanningTreeProtocol	
.BridgeIdentifier	Combination of dot1dStpPriority and dot1dBaseBridgeAddress
.TopologyChange	Since this is transitory, it is not considered useful.
SpanningTreeProtocolPort	
.Uptime	Same as ifLastChange (MIB II)
.PortIdentifier	Combination of dot1dStpPort and dot1dStpPortPriority
.TopologyChangeAcknowledged	Since this is transitory, it is not considered useful.
.DiscardLackOfBuffers	Redundant
Transmission Priority	These objects are not required as per the PICS Proforma and not considered useful.
.TransmissionPriorityName	
.OutboundUserPriority	
.OutboundAccessPriority	

3.1.1. The dotldBase Group

This mandatory group contains the objects which are applicable to all types of bridges.

3.1.2. The dotldStp Group

This group contains the objects that denote the bridge's state with respect to the Spanning Tree Protocol. If a node does not implement the Spanning Tree Protocol, this group will not be implemented.

3.1.3. The dotldSr Group

This group contains the objects that describe the entity's state with respect to source route bridging. If source routing is not supported this group will not be implemented. This group is applicable to source route only, and SRT bridges. This group will be described in a separate document applicable only to source route bridging.

3.1.4. The dotldTp Group

This group contains objects that describe the entity's state with respect to transparent bridging. If transparent bridging is not supported this group will not be implemented. This group is applicable to transparent only and SRT bridges.

3.1.5. The dotldStatic Group

This group contains objects that describe the entity's state with respect to destination-address filtering. If destination-address filtering is not supported this group will not be implemented. This group is applicable to any type of bridge which performs destination-address filtering.

3.2. Relationship to Other MIBs

As described above, some IEEE 802.1d management objects have not been included in this MIB because they overlap with objects in other MIBs applicable to a bridge implementing this MIB. In particular, it is assumed that a bridge implementing this MIB will also implement (at least) the 'system' group and the 'interfaces' group defined in MIB-II [6].

3.2.1. Relationship to the 'system' group

In MIB-II, the 'system' group is defined as being mandatory for all systems such that each managed entity contains one instance of each

object in the 'system' group. Thus, those objects apply to the entity as a whole irrespective of whether the entity's sole functionality is bridging, or whether bridging is only a subset of the entity's functionality.

3.2.2. Relationship to the 'interfaces' group

In MIB-II, the 'interfaces' group is defined as being mandatory for all systems and contains information on an entity's interfaces, where each interface is thought of as being attached to a 'subnetwork'. (Note that this term is not to be confused with 'subnet' which refers to an addressing partitioning scheme used in the Internet suite of protocols.) The term 'segment' is used in this memo to refer to such a subnetwork, whether it be an Ethernet segment, a 'ring', a WAN link, or even an X.25 virtual circuit.

Implicit in this Bridge MIB is the notion of ports on a bridge. Each of these ports is associated with one interface of the 'interfaces' group, and in most situations, each port is associated with a different interface. However, there are situations in which multiple ports are associated with the same interface. An example of such a situation would be several ports each corresponding one-to-one with several X.25 virtual circuits but all on the same interface.

Each port is uniquely identified by a port number. A port number has no mandatory relationship to an interface number, but in the simple case a port number will have the same value as the corresponding interface's interface number. Port numbers are in the range (1..dot1dBaseNumPorts).

Some entities perform other functionality as well as bridging through the sending and receiving of data on their interfaces. In such situations, only a subset of the data sent/received on an interface is within the domain of the entity's bridging functionality. This subset is considered to be delineated according to a set of protocols, with some protocols being bridged, and other protocols not being bridged. For example, in an entity which exclusively performed bridging, all protocols would be considered as being bridged, whereas in an entity which performed IP routing on IP datagrams and only bridged other protocols, only the non-IP data would be considered as being bridged.

Thus, this Bridge MIB (and in particular, its counters) are applicable only to that subset of the data on an entity's interfaces which is sent/received for a protocol being bridged. All such data is sent/received via the ports of the bridge.

3.3. Textual Conventions

The datatypes, MacAddress, BridgeId and Timeout, are used as textual conventions in this document. These textual conventions have NO effect on either the syntax nor the semantics of any managed object. Objects defined using these conventions are always encoded by means of the rules that define their primitive type. Hence, no changes to the SMI or the SNMP are necessary to accommodate these textual conventions which are adopted merely for the convenience of readers.

4. Changes from RFC 1286

- (1) Updated all text to remove references to source route bridging where not applicable. SR MIB will be a separate document.
- (2) Removed dotldSrPortTable. Retained OID definition of dotldSr.
- (3) Updated all references of "draft P802.1d/D9" to "IEEE 802.1D-1990".
- (4) Updated bibliography.
- (5) Added clarification to description of dotldPortPathCost.
- (6) Put recommended default in description of dotldStaticAllowedToGoTo.
- (7) Put recommended default in description of dotldStaticStatus.
- (8) Put recommended default in description of dotldTpAgingTime. Specified range of (10..1000000).
- (9) Updated all port number syntaxes, when used as index, to use the range (1..65535).
- (10) Updated definition of dotldTpPortInFrames and dotldTpPortOutFrames.
- (11) Added text to the traps indicating that they are optional.
- (12) Clarified definition of dotldStpForwardDelay.

5. Definitions

```
BRIDGE-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
    Counter, TimeTicks
        FROM RFC1155-SMI
    mib-2
        FROM RFC1213-MIB
    OBJECT-TYPE
        FROM RFC-1212
    TRAP-TYPE
        FROM RFC-1215;
```

```
-- All representations of MAC addresses in this MIB Module
-- use, as a textual convention (i.e. this convention does
-- not affect their encoding), the data type:
```

```
MacAddress ::= OCTET STRING (SIZE (6))    -- a 6 octet address
                                           -- in the
                                           -- "canonical"
                                           -- order
-- defined by IEEE 802.1a, i.e., as if it were transmitted
-- least significant bit first, even though 802.5 (in
-- contrast to other n802.x protocols) requires MAC
-- addresses to be transmitted most significant bit first.
--
-- 16-bit addresses, if needed, are represented by setting
-- their upper 4 octets to all 0's, i.e., AAFF would be
-- represented as 00000000AAFF.
```

```
-- Similarly, all representations of Bridge-Id in this MIB
-- Module use, as a textual convention (i.e. this
-- convention does not affect their encoding), the data
-- type:
```

```
BridgeId ::= OCTET STRING (SIZE (8))    -- the
                                           -- Bridge-Identifier
                                           -- as used in the
                                           -- Spanning Tree
-- Protocol to uniquely identify a bridge. Its first two
-- octets (in network byte order) contain a priority
-- value and its last 6 octets contain the MAC address
-- used to refer to a bridge in a unique fashion
-- (typically, the numerically smallest MAC address
-- of all ports on the bridge).
```

```
-- Several objects in this MIB module represent values of
-- timers used by the Spanning Tree Protocol.  In this
-- MIB, these timers have values in units of hundredths of
-- a second (i.e. 1/100 secs).
-- These timers, when stored in a Spanning Tree Protocol's
-- BPDU, are in units of 1/256 seconds.  Note, however,
-- that 802.1D-1990 specifies a settable granularity of
-- no more than 1 second for these timers.  To avoid
-- ambiguity, a data type is defined here as a textual
-- convention and all representation of these timers
-- in this MIB module are defined using this data type.  An
-- algorithm is also defined for converting between the
-- different units, to ensure a timer's value is not
-- distorted by multiple conversions.
-- The data type is:
```

```
Timeout ::= INTEGER -- a STP timer in units of 1/100 seconds
```

```
-- To convert a Timeout value into a value in units of
-- 1/256 seconds, the following algorithm should be used:
```

```
--      b = floor( (n * 256) / 100)
```

```
-- where:
```

```
--      floor = quotient [ignore remainder]
```

```
--      n is the value in 1/100 second units
```

```
--      b is the value in 1/256 second units
```

```
-- To convert the value from 1/256 second units back to
-- 1/100 seconds, the following algorithm should be used:
```

```
--      n = ceiling( (b * 100) / 256)
```

```
-- where:
```

```
--      ceiling = quotient [if remainder is 0], or
--                  quotient + 1 [if remainder is non-zero]
```

```
--      n is the value in 1/100 second units
```

```
--      b is the value in 1/256 second units
```

```
-- Note: it is important that the arithmetic operations are
-- done in the order specified (i.e., multiply first, divide
-- second).
```

```
dot1dBridge OBJECT IDENTIFIER ::= { mib-2 17 }
```

```
-- groups in the Bridge MIB

dotldBase      OBJECT IDENTIFIER ::= { dotldBridge 1 }

dotldStp       OBJECT IDENTIFIER ::= { dotldBridge 2 }

dotldSr        OBJECT IDENTIFIER ::= { dotldBridge 3 }
-- separately documented

dotldTp        OBJECT IDENTIFIER ::= { dotldBridge 4 }

dotldStatic    OBJECT IDENTIFIER ::= { dotldBridge 5 }


-- the dotldBase group

-- Implementation of the dotldBase group is mandatory for all
-- bridges.

dotldBaseBridgeAddress OBJECT-TYPE
    SYNTAX  MacAddress
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The MAC address used by this bridge when it must
        be referred to in a unique fashion.  It is
        recommended that this be the numerically smallest
        MAC address of all ports that belong to this
        bridge.  However it is only required to be unique.
        When concatenated with dotldStpPriority a unique
        BridgeIdentifier is formed which is used in the
        Spanning Tree Protocol."
    REFERENCE
        "IEEE 802.1D-1990: Sections 6.4.1.1.3 and 3.12.5"
    ::= { dotldBase 1 }

dotldBaseNumPorts OBJECT-TYPE
    SYNTAX  INTEGER
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The number of ports controlled by this bridging
        entity."
    REFERENCE
        "IEEE 802.1D-1990: Section 6.4.1.1.3"
    ::= { dotldBase 2 }

dotldBaseType OBJECT-TYPE
```

```

SYNTAX  INTEGER {
            unknown(1),
            transparent-only(2),
            sourceroute-only(3),
            srt(4)
        }
ACCESS   read-only
STATUS   mandatory
DESCRIPTION
    "Indicates what type of bridging this bridge can
    perform.  If a bridge is actually performing a
    certain type of bridging this will be indicated by
    entries in the port table for the given type."
 ::= { dot1dBase 3 }

-- The Generic Bridge Port Table

dot1dBasePortTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF Dot1dBasePortEntry
    ACCESS   not-accessible
    STATUS   mandatory
    DESCRIPTION
        "A table that contains generic information about
        every port that is associated with this bridge.
        Transparent, source-route, and srt ports are
        included."
    ::= { dot1dBase 4 }

dot1dBasePortEntry OBJECT-TYPE
    SYNTAX  Dot1dBasePortEntry
    ACCESS   not-accessible
    STATUS   mandatory
    DESCRIPTION
        "A list of information for each port of the
        bridge."
    REFERENCE
        "IEEE 802.1D-1990: Section 6.4.2, 6.6.1"
    INDEX   { dot1dBasePort }
    ::= { dot1dBasePortTable 1 }

Dot1dBasePortEntry ::=
    SEQUENCE {
        dot1dBasePort
            INTEGER,
        dot1dBasePortIfIndex
            INTEGER,
        dot1dBasePortCircuit
    }

```

```

        OBJECT IDENTIFIER,
        dotldBasePortDelayExceededDiscards
        Counter,
        dotldBasePortMtuExceededDiscards
        Counter
    }

dotldBasePort OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The port number of the port for which this entry
        contains bridge management information."
    ::= { dotldBasePortEntry 1 }

dotldBasePortIfIndex OBJECT-TYPE
    SYNTAX  INTEGER
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The value of the instance of the ifIndex object,
        defined in MIB-II, for the interface corresponding
        to this port."
    ::= { dotldBasePortEntry 2 }

dotldBasePortCircuit OBJECT-TYPE
    SYNTAX  OBJECT IDENTIFIER
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "For a port which (potentially) has the same value
        of dotldBasePortIfIndex as another port on the
        same bridge, this object contains the name of an
        object instance unique to this port.  For example,
        in the case where multiple ports correspond one-
        to-one with multiple X.25 virtual circuits, this
        value might identify an (e.g., the first) object
        instance associated with the X.25 virtual circuit
        corresponding to this port.

        For a port which has a unique value of
        dotldBasePortIfIndex, this object can have the
        value { 0 0 }."
    ::= { dotldBasePortEntry 3 }

dotldBasePortDelayExceededDiscards OBJECT-TYPE
    SYNTAX  Counter

```

```

ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "The number of frames discarded by this port due
    to excessive transit delay through the bridge. It
    is incremented by both transparent and source
    route bridges."
REFERENCE
    "IEEE 802.1D-1990: Section 6.6.1.1.3"
::= { dot1dBasePortEntry 4 }

```

dot1dBasePortMtuExceededDiscards OBJECT-TYPE

```

SYNTAX  Counter
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "The number of frames discarded by this port due
    to an excessive size. It is incremented by both
    transparent and source route bridges."
REFERENCE
    "IEEE 802.1D-1990: Section 6.6.1.1.3"
::= { dot1dBasePortEntry 5 }

```

-- the dot1dStp group

-- Implementation of the dot1dStp group is optional. It is
 -- implemented by those bridges that support the Spanning Tree
 -- Protocol.

dot1dStpProtocolSpecification OBJECT-TYPE

```

SYNTAX  INTEGER {
            unknown(1),
            decLb100(2),
            ieee8021d(3)
        }
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "An indication of what version of the Spanning
    Tree Protocol is being run. The value
    'decLb100(2)' indicates the DEC LANbridge 100
    Spanning Tree protocol. IEEE 802.1d
    implementations will return 'ieee8021d(3)'. If
    future versions of the IEEE Spanning Tree Protocol
    are released that are incompatible with the
    current version a new value will be defined."

```

```
::= { dot1dStp 1 }
```

dot1dStpPriority OBJECT-TYPE

SYNTAX INTEGER (0..65535)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The value of the write-able portion of the Bridge ID, i.e., the first two octets of the (8 octet long) Bridge ID. The other (last) 6 octets of the Bridge ID are given by the value of dot1dBaseBridgeAddress."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.7"

```
::= { dot1dStp 2 }
```

dot1dStpTimeSinceTopologyChange OBJECT-TYPE

SYNTAX TimeTicks

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The time (in hundredths of a second) since the last time a topology change was detected by the bridge entity."

REFERENCE

"IEEE 802.1D-1990: Section 6.8.1.1.3"

```
::= { dot1dStp 3 }
```

dot1dStpTopChanges OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The total number of topology changes detected by this bridge since the management entity was last reset or initialized."

REFERENCE

"IEEE 802.1D-1990: Section 6.8.1.1.3"

```
::= { dot1dStp 4 }
```

dot1dStpDesignatedRoot OBJECT-TYPE

SYNTAX BridgeId

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The bridge identifier of the root of the spanning tree as determined by the Spanning Tree Protocol as executed by this node. This value is used as

the Root Identifier parameter in all Configuration Bridge PDUs originated by this node."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.1"

::= { dot1dStp 5 }

dot1dStpRootCost OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The cost of the path to the root as seen from this bridge."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.2"

::= { dot1dStp 6 }

dot1dStpRootPort OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The port number of the port which offers the lowest cost path from this bridge to the root bridge."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.3"

::= { dot1dStp 7 }

dot1dStpMaxAge OBJECT-TYPE

SYNTAX Timeout

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The maximum age of Spanning Tree Protocol information learned from the network on any port before it is discarded, in units of hundredths of a second. This is the actual value that this bridge is currently using."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.4"

::= { dot1dStp 8 }

dot1dStpHelloTime OBJECT-TYPE

SYNTAX Timeout

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The amount of time between the transmission of Configuration bridge PDUs by this node on any port when it is the root of the spanning tree or trying to become so, in units of hundredths of a second. This is the actual value that this bridge is currently using."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.5"

::= { dot1dStp 9 }

dot1dStpHoldTime OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

"This time value determines the interval length during which no more than two Configuration bridge PDUs shall be transmitted by this node, in units of hundredths of a second."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.14"

::= { dot1dStp 10 }

dot1dStpForwardDelay OBJECT-TYPE

SYNTAX Timeout

ACCESS read-only

STATUS mandatory

DESCRIPTION

"This time value, measured in units of hundredths of a second, controls how fast a port changes its spanning state when moving towards the Forwarding state. The value determines how long the port stays in each of the Listening and Learning states, which precede the Forwarding state. This value is also used, when a topology change has been detected and is underway, to age all dynamic entries in the Forwarding Database. [Note that this value is the one that this bridge is currently using, in contrast to dot1dStpBridgeForwardDelay which is the value that this bridge and all others would start using if/when this bridge were to become the root.]"

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.6"

::= { dot1dStp 11 }

dot1dStpBridgeMaxAge OBJECT-TYPE

SYNTAX Timeout (600..4000)

ACCESS read-write
STATUS mandatory
DESCRIPTION

"The value that all bridges use for MaxAge when this bridge is acting as the root. Note that 802.1D-1990 specifies that the range for this parameter is related to the value of dot1dStpBridgeHelloTime. The granularity of this timer is specified by 802.1D-1990 to be 1 second. An agent may return a badValue error if a set is attempted to a value which is not a whole number of seconds."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.8"

::= { dot1dStp 12 }

dot1dStpBridgeHelloTime OBJECT-TYPE

SYNTAX Timeout (100..1000)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The value that all bridges use for HelloTime when this bridge is acting as the root. The granularity of this timer is specified by 802.1D-1990 to be 1 second. An agent may return a badValue error if a set is attempted to a value which is not a whole number of seconds."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.9"

::= { dot1dStp 13 }

dot1dStpBridgeForwardDelay OBJECT-TYPE

SYNTAX Timeout (400..3000)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The value that all bridges use for ForwardDelay when this bridge is acting as the root. Note that 802.1D-1990 specifies that the range for this parameter is related to the value of dot1dStpBridgeMaxAge. The granularity of this timer is specified by 802.1D-1990 to be 1 second. An agent may return a badValue error if a set is attempted to a value which is not a whole number of seconds."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.3.10"

::= { dot1dStp 14 }

-- The Spanning Tree Port Table

```
dot1dStpPortTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF Dot1dStpPortEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "A table that contains port-specific information
        for the Spanning Tree Protocol."
    ::= { dot1dStp 15 }
```

```
dot1dStpPortEntry OBJECT-TYPE
    SYNTAX  Dot1dStpPortEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "A list of information maintained by every port
        about the Spanning Tree Protocol state for that
        port."
    INDEX   { dot1dStpPort }
    ::= { dot1dStpPortTable 1 }
```

```
Dot1dStpPortEntry ::=
    SEQUENCE {
        dot1dStpPort
            INTEGER,
        dot1dStpPortPriority
            INTEGER,
        dot1dStpPortState
            INTEGER,
        dot1dStpPortEnable
            INTEGER,
        dot1dStpPortPathCost
            INTEGER,
        dot1dStpPortDesignatedRoot
            BridgeId,
        dot1dStpPortDesignatedCost
            INTEGER,
        dot1dStpPortDesignatedBridge
            BridgeId,
        dot1dStpPortDesignatedPort
            OCTET STRING,
        dot1dStpPortForwardTransitions
            Counter
    }
```

```
dot1dStpPort OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
```

ACCESS read-only
STATUS mandatory
DESCRIPTION
 "The port number of the port for which this entry
 contains Spanning Tree Protocol management
 information."
REFERENCE
 "IEEE 802.1D-1990: Section 6.8.2.1.2"
::= { dot1dStpPortEntry 1 }

dot1dStpPortPriority OBJECT-TYPE
SYNTAX INTEGER (0..255)
ACCESS read-write
STATUS mandatory
DESCRIPTION
 "The value of the priority field which is
 contained in the first (in network byte order)
 octet of the (2 octet long) Port ID. The other
 octet of the Port ID is given by the value of
 dot1dStpPort."
REFERENCE
 "IEEE 802.1D-1990: Section 4.5.5.1"
::= { dot1dStpPortEntry 2 }

dot1dStpPortState OBJECT-TYPE
SYNTAX INTEGER {
 disabled(1),
 blocking(2),
 listening(3),
 learning(4),
 forwarding(5),
 broken(6)
}
ACCESS read-only
STATUS mandatory
DESCRIPTION
 "The port's current state as defined by
 application of the Spanning Tree Protocol. This
 state controls what action a port takes on
 reception of a frame. If the bridge has detected
 a port that is malfunctioning it will place that
 port into the broken(6) state. For ports which
 are disabled (see dot1dStpPortEnable), this object
 will have a value of disabled(1)."
REFERENCE
 "IEEE 802.1D-1990: Section 4.5.5.2"
::= { dot1dStpPortEntry 3 }

dot1dStpPortEnable OBJECT-TYPE

```
SYNTAX  INTEGER {
            enabled(1),
            disabled(2)
        }
```

```
ACCESS  read-write
```

```
STATUS  mandatory
```

DESCRIPTION

"The enabled/disabled status of the port."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.5.2"

```
::= { dot1dStpPortEntry 4 }
```

dot1dStpPortPathCost OBJECT-TYPE

```
SYNTAX  INTEGER (1..65535)
```

```
ACCESS  read-write
```

```
STATUS  mandatory
```

DESCRIPTION

"The contribution of this port to the path cost of paths towards the spanning tree root which include this port. 802.1D-1990 recommends that the default value of this parameter be in inverse proportion to the speed of the attached LAN."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.5.3"

```
::= { dot1dStpPortEntry 5 }
```

dot1dStpPortDesignatedRoot OBJECT-TYPE

```
SYNTAX  BridgeId
```

```
ACCESS  read-only
```

```
STATUS  mandatory
```

DESCRIPTION

"The unique Bridge Identifier of the Bridge recorded as the Root in the Configuration BPDUs transmitted by the Designated Bridge for the segment to which the port is attached."

REFERENCE

"IEEE 802.1D-1990: Section 4.5.5.4"

```
::= { dot1dStpPortEntry 6 }
```

dot1dStpPortDesignatedCost OBJECT-TYPE

```
SYNTAX  INTEGER
```

```
ACCESS  read-only
```

```
STATUS  mandatory
```

DESCRIPTION

"The path cost of the Designated Port of the segment connected to this port. This value is compared to the Root Path Cost field in received

```

        bridge PDUs."
REFERENCE
    "IEEE 802.1D-1990: Section 4.5.5.5"
 ::= { dot1dStpPortEntry 7 }

dot1dStpPortDesignatedBridge OBJECT-TYPE
    SYNTAX  BridgeId
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The Bridge Identifier of the bridge which this
         port considers to be the Designated Bridge for
         this port's segment."
    REFERENCE
        "IEEE 802.1D-1990: Section 4.5.5.6"
 ::= { dot1dStpPortEntry 8 }

dot1dStpPortDesignatedPort OBJECT-TYPE
    SYNTAX  OCTET STRING (SIZE (2))
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The Port Identifier of the port on the Designated
         Bridge for this port's segment."
    REFERENCE
        "IEEE 802.1D-1990: Section 4.5.5.7"
 ::= { dot1dStpPortEntry 9 }

dot1dStpPortForwardTransitions OBJECT-TYPE
    SYNTAX  Counter
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The number of times this port has transitioned
         from the Learning state to the Forwarding state."
 ::= { dot1dStpPortEntry 10 }

-- the dot1dTp group

-- Implementation of the dot1dTp group is optional.  It is
-- implemented by those bridges that support the transparent
-- bridging mode.  A transparent or SRT bridge will implement
-- this group.

dot1dTpLearnedEntryDiscards OBJECT-TYPE
    SYNTAX  Counter

```

ACCESS read-only
 STATUS mandatory
 DESCRIPTION

"The total number of Forwarding Database entries, which have been or would have been learnt, but have been discarded due to a lack of space to store them in the Forwarding Database. If this counter is increasing, it indicates that the Forwarding Database is regularly becoming full (a condition which has unpleasant performance effects on the subnetwork). If this counter has a significant value but is not presently increasing, it indicates that the problem has been occurring but is not persistent."

REFERENCE

"IEEE 802.1D-1990: Section 6.7.1.1.3"

::= { dot1dTp 1 }

dot1dTpAgingTime OBJECT-TYPE

SYNTAX INTEGER (10..1000000)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The timeout period in seconds for aging out dynamically learned forwarding information.

802.1D-1990 recommends a default of 300 seconds."

REFERENCE

"IEEE 802.1D-1990: Section 6.7.1.1.3"

::= { dot1dTp 2 }

-- The Forwarding Database for Transparent Bridges

dot1dTpFdbTable OBJECT-TYPE

SYNTAX SEQUENCE OF Dot1dTpFdbEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"A table that contains information about unicast entries for which the bridge has forwarding and/or filtering information. This information is used by the transparent bridging function in determining how to propagate a received frame."

::= { dot1dTp 3 }

dot1dTpFdbEntry OBJECT-TYPE

SYNTAX Dot1dTpFdbEntry

ACCESS not-accessible

```

STATUS    mandatory
DESCRIPTION
    "Information about a specific unicast MAC address
    for which the bridge has some forwarding and/or
    filtering information."
INDEX     { dot1dTpFdbAddress }
::= { dot1dTpFdbTable 1 }

Dot1dTpFdbEntry ::=
SEQUENCE {
    dot1dTpFdbAddress
        MacAddress,
    dot1dTpFdbPort
        INTEGER,
    dot1dTpFdbStatus
        INTEGER
}

dot1dTpFdbAddress OBJECT-TYPE
SYNTAX    MacAddress
ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "A unicast MAC address for which the bridge has
    forwarding and/or filtering information."
REFERENCE
    "IEEE 802.1D-1990: Section 3.9.1, 3.9.2"
::= { dot1dTpFdbEntry 1 }

dot1dTpFdbPort OBJECT-TYPE
SYNTAX    INTEGER
ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "Either the value '0', or the port number of the
    port on which a frame having a source address
    equal to the value of the corresponding instance
    of dot1dTpFdbAddress has been seen.  A value of
    '0' indicates that the port number has not been
    learned but that the bridge does have some
    forwarding/filtering information about this
    address (e.g. in the dot1dStaticTable).
    Implementors are encouraged to assign the port
    value to this object whenever it is learned even
    for addresses for which the corresponding value of
    dot1dTpFdbStatus is not learned(3)."
::= { dot1dTpFdbEntry 2 }

```


dot1dTpFdbStatus OBJECT-TYPE

```
SYNTAX  INTEGER {
    other(1),
    invalid(2),
    learned(3),
    self(4),
    mgmt(5)
}
```

```
ACCESS  read-only
```

```
STATUS  mandatory
```

DESCRIPTION

"The status of this entry. The meanings of the values are:

other(1) : none of the following. This would include the case where some other MIB object (not the corresponding instance of dot1dTpFdbPort, nor an entry in the dot1dStaticTable) is being used to determine if and how frames addressed to the value of the corresponding instance of dot1dTpFdbAddress are being forwarded.

invalid(2) : this entry is not longer valid (e.g., it was learned but has since aged-out), but has not yet been flushed from the table.

learned(3) : the value of the corresponding instance of dot1dTpFdbPort was learned, and is being used.

self(4) : the value of the corresponding instance of dot1dTpFdbAddress represents one of the bridge's addresses. The corresponding instance of dot1dTpFdbPort indicates which of the bridge's ports has this address.

mgmt(5) : the value of the corresponding instance of dot1dTpFdbAddress is also the value of an existing instance of dot1dStaticAddress."

```
::= { dot1dTpFdbEntry 3 }
```

-- Port Table for Transparent Bridges

```
dotldTpPortTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF DotldTpPortEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "A table that contains information about every
        port that is associated with this transparent
        bridge."
    ::= { dotldTp 4 }
```

```
dotldTpPortEntry OBJECT-TYPE
    SYNTAX  DotldTpPortEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "A list of information for each port of a
        transparent bridge."
    INDEX   { dotldTpPort }
    ::= { dotldTpPortTable 1 }
```

```
DotldTpPortEntry ::=
    SEQUENCE {
        dotldTpPort
            INTEGER,
        dotldTpPortMaxInfo
            INTEGER,
        dotldTpPortInFrames
            Counter,
        dotldTpPortOutFrames
            Counter,
        dotldTpPortInDiscards
            Counter
    }
```

```
dotldTpPort OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The port number of the port for which this entry
        contains Transparent bridging management
        information."
    ::= { dotldTpPortEntry 1 }
```

-- It would be nice if we could use ifMtu as the size of the
 -- largest INFO field, but we can't because ifMtu is defined

-- to be the size that the (inter-)network layer can use which
-- can differ from the MAC layer (especially if several layers
-- of encapsulation are used).

dot1dTpPortMaxInfo OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The maximum size of the INFO (non-MAC) field that
this port will receive or transmit."

::= { dot1dTpPortEntry 2 }

dot1dTpPortInFrames OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of frames that have been received by
this port from its segment. Note that a frame
received on the interface corresponding to this
port is only counted by this object if and only if
it is for a protocol being processed by the local
bridging function, including bridge management
frames."

REFERENCE

"IEEE 802.1D-1990: Section 6.6.1.1.3"

::= { dot1dTpPortEntry 3 }

dot1dTpPortOutFrames OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of frames that have been transmitted
by this port to its segment. Note that a frame
transmitted on the interface corresponding to this
port is only counted by this object if and only if
it is for a protocol being processed by the local
bridging function, including bridge management
frames."

REFERENCE

"IEEE 802.1D-1990: Section 6.6.1.1.3"

::= { dot1dTpPortEntry 4 }

dot1dTpPortInDiscards OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory
DESCRIPTION
"Count of valid frames received which were
discarded (i.e., filtered) by the Forwarding
Process."
REFERENCE
"IEEE 802.1D-1990: Section 6.6.1.1.3"
::= { dot1dTpPortEntry 5 }

-- The Static (Destination-Address Filtering) Database

-- Implementation of this group is optional.

dot1dStaticTable OBJECT-TYPE
SYNTAX SEQUENCE OF Dot1dStaticEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"A table containing filtering information
configured into the bridge by (local or network)
management specifying the set of ports to which
frames received from specific ports and containing
specific destination addresses are allowed to be
forwarded. The value of zero in this table as the
port number from which frames with a specific
destination address are received, is used to
specify all ports for which there is no specific
entry in this table for that particular
destination address. Entries are valid for
unicast and for group/broadcast addresses."
REFERENCE
"IEEE 802.1D-1990: Section 6.7.2"
::= { dot1dStatic 1 }

dot1dStaticEntry OBJECT-TYPE
SYNTAX Dot1dStaticEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"Filtering information configured into the bridge
by (local or network) management specifying the
set of ports to which frames received from a
specific port and containing a specific
destination address are allowed to be forwarded."
REFERENCE
"IEEE 802.1D-1990: Section 6.7.2"

```
INDEX    { dotldStaticAddress, dotldStaticReceivePort }
 ::= { dotldStaticTable 1 }

DotldStaticEntry ::=
  SEQUENCE {
    dotldStaticAddress
      MacAddress,
    dotldStaticReceivePort
      INTEGER,
    dotldStaticAllowedToGoTo
      OCTET STRING,
    dotldStaticStatus
      INTEGER
  }

dotldStaticAddress OBJECT-TYPE
  SYNTAX  MacAddress
  ACCESS  read-write
  STATUS  mandatory
  DESCRIPTION
    "The destination MAC address in a frame to which
     this entry's filtering information applies.  This
     object can take the value of a unicast address, a
     group address or the broadcast address."
  REFERENCE
    "IEEE 802.1D-1990: Section 3.9.1, 3.9.2"
  ::= { dotldStaticEntry 1 }

dotldStaticReceivePort OBJECT-TYPE
  SYNTAX  INTEGER
  ACCESS  read-write
  STATUS  mandatory
  DESCRIPTION
    "Either the value '0', or the port number of the
     port from which a frame must be received in order
     for this entry's filtering information to apply.
     A value of zero indicates that this entry applies
     on all ports of the bridge for which there is no
     other applicable entry."
  ::= { dotldStaticEntry 2 }

dotldStaticAllowedToGoTo OBJECT-TYPE
  SYNTAX  OCTET STRING
  ACCESS  read-write
  STATUS  mandatory
  DESCRIPTION
    "The set of ports to which frames received from a
     specific port and destined for a specific MAC
```

address, are allowed to be forwarded. Each octet within the value of this object specifies a set of eight ports, with the first octet specifying ports 1 through 8, the second octet specifying ports 9 through 16, etc. Within each octet, the most significant bit represents the lowest numbered port, and the least significant bit represents the highest numbered port. Thus, each port of the bridge is represented by a single bit within the value of this object. If that bit has a value of '1' then that port is included in the set of ports; the port is not included if its bit has a value of '0'. (Note that the setting of the bit corresponding to the port from which a frame is received is irrelevant.) The default value of this object is a string of ones of appropriate length."

```
::= { dot1dStaticEntry 3 }
```

dot1dStaticStatus OBJECT-TYPE

```
SYNTAX  INTEGER {
    other(1),
    invalid(2),
    permanent(3),
    deleteOnReset(4),
    deleteOnTimeout(5)
}
```

ACCESS read-write

STATUS mandatory

DESCRIPTION

"This object indicates the status of this entry. The default value is permanent(3).

other(1) - this entry is currently in use but the conditions under which it will remain so are different from each of the following values.

invalid(2) - writing this value to the object removes the corresponding entry.

permanent(3) - this entry is currently in use and will remain so after the next reset of the bridge.

deleteOnReset(4) - this entry is currently in use and will remain so until the next reset of the bridge.

deleteOnTimeout(5) - this entry is currently in use and will remain so until it is aged out."

```
 ::= { dot1dStaticEntry 4 }

-- Traps for use by Bridges

-- Traps for the Spanning Tree Protocol

newRoot TRAP-TYPE
    ENTERPRISE dot1dBridge
    DESCRIPTION
        "The newRoot trap indicates that the sending agent
        has become the new root of the Spanning Tree; the
        trap is sent by a bridge soon after its election
        as the new root, e.g., upon expiration of the
        Topology Change Timer immediately subsequent to
        its election. Implementation of this trap is
        optional."
    ::= 1

topologyChange TRAP-TYPE
    ENTERPRISE dot1dBridge
    DESCRIPTION
        "A topologyChange trap is sent by a bridge when
        any of its configured ports transitions from the
        Learning state to the Forwarding state, or from
        the Forwarding state to the Blocking state. The
        trap is not sent if a newRoot trap is sent for the
        same transition. Implementation of this trap is
        optional."
    ::= 2

END
```

6. Acknowledgments

This document was produced on behalf of the Bridge Sub-Working Group of the SNMP Working Group of the Internet Engineering Task Force. Over the course of its deliberations, the working group received four separate documents for consideration as the basis for its work. The first was submitted by Stan Froyd of Advanced Computer Communications; the second by Richard Fox of SynOptics; the third by Eric Decker of cisco Inc. and Keith McCloghrie of Hughes LAN Systems; and the fourth by Paul Langille and Anil Rijssinghani of Digital Equipment Corp. After considering the submissions, the working group chose to proceed with a document formed as a conjunction of the latter two submissions. This document is the result.

The authors wish to thank the members of the Bridge Working Group for their many comments and suggestions which improved this effort. In particular, Fred Baker (chairman of the working group) of ACC, Steve Sherry of Xyplex, and Frank Kastenholz of Clearpoint Research Corp. Others members of the Bridge Working Group who contributed to this effort are:

Bill Anderson, Mitre
Karl Auerbach, Epilogue
Fred Baker, ACC (chair)
Terry Bradley, Wellfleet
Ted Brunner, Bellcore
Jeffrey Buffum, Apollo
Chris ChioTasso, Fibronics
Anthony Chung, HLS
Chuck Davin, MIT-LCS
Andy Davis, Spider
Eric Decker, cisco
Nadya El-Afandi, Network Systems
Gary Ellis, HP/Apollo
Richard Fox, SynOptics
Stan Froyd, ACC
Frank Kastenholz, Clearpoint Research
Shirnshon Kaufman,
Jim Kinder, Fibercom
Cheryl Krupczak, NCR
Paul Langille, Digital
Peter Lin, Vitalink
Keith McCloghrie, HLS
Donna McMaster, SynOptics
Dave Perkins, 3Com
Jim Reinstedler, Ungermann Bass
Anil Rijsinghani, Digital
Mark Schaefer, David Systems
Steve Sherry, Xyplex
Bob Stewart, Xyplex
Emil Sturniolo,
Kevin Synott, Retix
Ian Thomas, Chipcom
Maurice Turcott, Racal
Fei Xu,

7. References

- [1] Cerf, V., "IAB Recommendations for the Development of Internet Network Management Standards", RFC 1052, NRI, April 1988.
- [2] Cerf, V., "Report of the Second Ad Hoc Network Management Review Group", RFC 1109, NRI, August 1989.
- [3] Rose M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based internets", STD 16, RFC 1155, Performance Systems International, Hughes LAN Systems, May 1990.
- [4] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May 1990.
- [5] McCloghrie K., and M. Rose, Editors, "Management Information Base for Network Management of TCP/IP-based internets", STD 17, RFC 1213, Performance Systems International, March 1991.
- [6] Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1), International Organization for Standardization, International Standard 8824, December 1987.
- [7] Information processing systems - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Notation One (ASN.1), International Organization for Standardization, International Standard 8825, December 1987.
- [8] Rose, M., and K. McCloghrie, Editors, "Concise MIB Definitions", STD 16, RFC 1212, Performance Systems International, Hughes LAN Systems, March 1991.
- [9] Rose, M., Editor, "A Convention for Defining Traps for use with the SNMP", RFC 1215, Performance Systems International, March 1991.
- [10] ANSI/IEEE Standard 802.1D-1990 MAC Bridges, IEEE Project 802 Local and Metropolitan Area Networks, (March 8, 1991).
- [11] ISO DIS 10038 MAC Bridges.

8. Security Considerations

Security issues are not discussed in this memo.

9. Authors' Addresses

Eric B. Decker
cisco Systems, Inc.
1525 O'Brien Dr.
Menlo Park, CA 94025

Phone: (415) 326-1941
Email: cire@cisco.com

Paul Langille
Digital Equipment Corporation
Digital Drive, MK02-2/K03
Merrimack, NH 03054

Phone: (603) 884-4045
EMail: langille@edwin.enet.dec.com

Anil Rijsinghani
Digital Equipment Corporation
550 King Street
Littleton, MA 01460

Phone: (508) 486-6786
EMail: anil@levers.enet.dec.com

Keith McCloghrie
Hughes LAN Systems, Inc.
1225 Charleston Road
Mountain View, CA 94043

Phone: (415) 966-7934
EMail: kzm@hls.com