Dial Control Management Information Base using SMIv2

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) 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 the Internet community. In particular, it describes managed objects used for managing demand access circuits, including ISDN.

This document specifies a MIB module in a manner that is compliant to the SNMPv2 SMI. The set of objects is consistent with the SNMP framework and existing SNMP standards.

This document is a product of the ISDN MIB working group within the Internet Engineering Task Force. Comments are solicited and should be addressed to the working group’s mailing list at isdn-mib@cisco.com and/or the author.

Table of Contents

1 The SNMPv2 Network Management Framework ...................... 2
1.1 Object Definitions ........................................ 2
2 Overview .................................................... 2
2.1 Structure of MIB ........................................ 2
2.2 Relationship to the Interfaces MIB ........................ 3
2.2.1 Layering Model and Virtual Circuits .............................. 3
2.2.2 ifTestTable ........................................... 4
2.2.3 ifRcvAddressTable ......................................... 4
2.2.3.1 ifEntry for a single peer ........................... 5
2.3 Multilink and backup line support ................................ 5
2.4 Support for generic peers .................................. 5
3 Definitions ................................................. 6
3.1 Dial Control MIB ........................................ 6
4 Acknowledgments ............................................. 32
5 References ................................................ 33

Roeck Standards Track [Page 1]
1. The SNMPv2 Network Management Framework

The SNMPv2 Network Management Framework presently consists of three major components. They are:

- the SMI, described in RFC 1902 [1] - the mechanisms used for describing and naming objects for the purpose of management.
- the protocol, STD 15, RFC 1157 [3] and/or RFC 1905 [4], - the protocol for accessing managed objects.

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

1.1. Object Definitions

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) defined in the SMI. In particular, each object type is named by an OBJECT IDENTIFIER, an administratively assigned name. 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 refer to the object type.

2. Overview

2.1. Structure of MIB

Managing demand access circuits requires the following groups of information:

- General configuration information.
- Information to describe peer configuration and peer statistics. In this respect, peer configuration means information on how to connect to peers on outgoing calls, how to identify peers on incoming calls, and other call related configuration information.
- Information to store active call information.
Information to retain call history.

The MIB, therefore, is structured into four groups:

- The dialCtlConfiguration group is used to specify general configuration information.
- The dialCtlPeer group is used to describe peer configuration and peer statistics.
- The callActive group is used to store active call information.
- The callHistory group is used to store call history information. These calls could be circuit switched or they could be virtual circuits. History of each and every call is stored, of successful calls as well as unsuccessful and rejected calls. An entry will be created when a call is cleared.

2.2. Relationship to the Interfaces MIB

This section clarifies the relationship of this MIB to the Interfaces MIB [8]. Several areas of correlation are addressed in the following subsections. The implementor is referred to the Interfaces MIB document in order to understand the general intent of these areas.

2.2.1. Layering Model and Virtual Circuits

On an occasional access channel, there are a number of peer systems that are permitted to call or be called, all of which need to be treated as active from a routing viewpoint, but most of which have no call in progress at any given time.

On dialup interfaces, this is further complicated by the fact that calls to a given peer float from channel to channel. One cannot definitively say "I call this peer on that interface." It is necessary, therefore, to provide a mapping algorithm between the low-level interfaces, and the various logical interfaces supporting the peers. This is solved by creating a logical interface (ifEntry) for each peer and a logical interface (ifEntry) for each low-level interface. These are then correlated using the ifStackTable.

The low-level interfaces are either physical interfaces, e.g. modem interfaces, or logical interfaces, e.g. ISDN B channels, which then in turn are layered on top of physical ISDN interfaces.
The model, therefore, looks something like this, taking ISDN as an example:

```
+-------------------------------------------------------+
|               Network Layer Protocol                  |
+------+ +-------+ +-------+ +-------+ +-------+ +------+
| |       | |       | |       | |       | | <== appears active
|PPP |   |PPP |   | F/R   |PPP |   |F/R   |
| for |   | for |   | for    | for |   |ifEntry with
| Peer1|   |Peer2|   | switch |Peer3|   |switch |
|     |   |     |   | A      |     |   |B     |
|+-+ +-+ +-+ +-+ +-+ +-+ +-+ +-+ +-+ +-+ +-+ +-+ +-+ +-+ 
|---+++----+++----+++----+++----+++----+++----+++----+++----+++---
|   B   | |   B   | |   B   | |   B   |
+--+ +--+ +--+ +--+ +--+ +--+ +--+ +--+ +--+ +--+ +--+ +--+ +--+
|   |   |   |   |   |   |   |   |   |   |   |   |   |
+-------------------------------------------------------+
```

Mapping of IP interfaces to Called Peers to B Channels

IfEntries are maintained for each peer.

In this model, each peer is required to have an associated encapsulation layer interface. This interface can be of any kind, e.g. PPP or LAPB.

In order to specify the network address for a given peer, one would then usually add a routing/forwarding table entry, pointing to the encapsulation layer interface through which this peer can be reached.

### 2.2.2. ifTestTable

The ifTestTable usage is defined in the MIBs defining the encapsulation below the network layer. For example, if PPP encapsulation is being used, the ifTestTable is defined by PPP.

### 2.2.3. ifRcvAddressTable

The ifRcvAddressTable usage is defined in the MIBs defining the encapsulation below the network layer. For example, if PPP encapsulation is being used, the ifRcvAddressTable is defined by PPP.
2.2.3.1. ifEntry for a single peer

IfEntries are defined in the MIBs defining the encapsulation below the network layer. For example, if PPP encapsulation is being used, the ifEntry is defined by PPP.

ifEntries will never be created by the Dial Control MIB. The Dial Control MIB always depends on some other ifIndex of some set of ifTypes. That is, to create an entry in the Dial Control MIB, the base ifEntry must already have been created through some other mechanism.

The Dial Control entry does have its own RowStatus, permitting the Dial Control supplementary information to come and go, but not otherwise disturbing the ifIndex to which it is attached. If in a given implementation the two are tightly bound, deleting the ifEntry may have the side effect of deleting the Dial Control entry.

2.3. Multilink and backup line support

In order to support multilink and backup procedures, there may be several entries for a single peer in the dialCtlPeerCfgTable.

A single peer is identified using the dialCtlPeerCfgId object of the dialCtlPeerCfgTable. There may be several entries in dialCtlPeerCfgTable with the same value of dialCtlPeerCfgId, but different ifIndex values. Each of those entries will then describe a possible connection to the same peer. Such entries can then be used to handle multilink as well as backup procedures, e.g. by bundling the attached ifEntries using PPP multilink.

2.4. Support for generic peers

Generic peers can for example be supported by permitting wild-card characters (e.g., ‘?’ or ‘**) in dialCtlPeerCfgAnswerAddress. A number to be accepted could then be defined as partly (e.g., ‘*1234’) or entirely generic (e.g., ‘**’).

A detailed specification of such a functionality is outside the scope of this document.

However, the implementor should be aware that supporting generic peers may cause a security hole. The user would not know where a call is from, which could potentially allow unauthorized access.
3. Definitions

3.1. Dial Control MIB

DIAL-CONTROL-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY,
    NOTIFICATION-TYPE,
    OBJECT-TYPE,
    Unsigned32
        FROM SNMPv2-SMI
    TEXTUAL-CONVENTION,
    DisplayString,
    TimeStamp,
    RowStatus
        FROM SNMPv2-TC
    MODULE-COMPLIANCE,
    OBJECT-GROUP,
    NOTIFICATION-GROUP
        FROM SNMPv2-CONF
    IANAifType
        FROM IANAifType-MIB
    ifOperStatus,
    ifIndex,
    InterfaceIndex,
    InterfaceIndexOrZero
        FROM IF-MIB
    transmission
        FROM RFC1213-MIB;

DIAL-CONTROL-MIB MODULE-IDENTITY
    LAST-UPDATED    "9609231544Z" -- Sep 23, 1996
    ORGANIZATION    "IETF ISDN Working Group"
    CONTACT-INFO
        " Guenter Roeck
        Postal: cisco Systems
                170 West Tasman Drive
                San Jose, CA 95134
                U.S.A.
        Phone: +1 408 527 3143
        E-mail: groeck@cisco.com"
    DESCRIPTION
        "The MIB module to describe peer information for
         demand access and possibly other kinds of interfaces."
    ::= { transmission 21 }

AbsoluteCounter32 ::= TEXTUAL-CONVENTION

Roeck Standards Track [Page 6]
STATUS       current
DESCRIPTION
"Represents a Counter32-like value that starts at zero,
does not decrease, and does not wrap. This may be used
only in situations where wrapping is not possible or
extremely unlikely. Should such a counter overflow,
it locks at the maximum value of 4,294,967,295.

The primary use of this type of counter is situations
where a counter value is to be recorded as history
and is thus no longer subject to reading for changing
values."

SYNTAX       Unsigned32

-- Dial Control Mib objects definitions
dialControlMibObjects OBJECT IDENTIFIER ::= { dialControlMib 1 }

-- General configuration group
dialCtlConfiguration OBJECT IDENTIFIER ::= { dialControlMibObjects 1 }

-- general configuration data/parameters
dialCtlAcceptMode OBJECT-TYPE
SYNTAX INTEGER {
    acceptNone(1),
    acceptAll(2),
    acceptKnown(3)
}
MAX-ACCESS  read-write
STATUS       current
DESCRIPTION
"The security level for acceptance of incoming calls.
acceptNone(1)  - incoming calls will not be accepted
acceptAll(2)   - incoming calls will be accepted,
even if there is no matching entry
in the dialCtlPeerCfgTable
acceptKnown(3) - incoming calls will be accepted only
if there is a matching entry in the
dialCtlPeerCfgTable
"

::= { dialCtlConfiguration 1 }
dialCtlTrapEnable OBJECT-TYPE
SYNTAX       INTEGER {
    enabled(1),
    disabled(2)
This object indicates whether dialCtlPeerCallInformation and dialCtlPeerCallSetup traps should be generated for all peers. If the value of this object is enabled(1), traps will be generated for all peers. If the value of this object is disabled(2), traps will be generated only for peers having dialCtlPeerCfgTrapEnable set to enabled(1).

DEFVAL { disabled } ::= { dialCtlConfiguration 2 }

-- Peer group
dialCtlPeer OBJECT IDENTIFIER ::= { dialControlMibObjects 2 }

-- peer configuration table
dialCtlPeerCfgTable OBJECT-TYPE
SYNTAX      SEQUENCE OF DialCtlPeerCfgEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
 "The list of peers from which the managed device will accept calls or to which it will place them."
 ::= { dialCtlPeer 1 }

dialCtlPeerCfgEntry OBJECT-TYPE
SYNTAX      DialCtlPeerCfgEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
 "Configuration data for a single Peer. This entry is effectively permanent, and contains information to identify the peer, how to connect to the peer, how to identify the peer and its permissions. The value of dialCtlPeerCfgOriginateAddress must be specified before a new row in this table can become active(1). Any writeable parameters in an existing entry can be modified while the entry is active. The modification will take effect when the peer in question will be called the next time. An entry in this table can only be created if the associated ifEntry already exists."
INDEX       { dialCtlPeerCfgId, ifIndex }
DialCtlPeerCfgEntry ::= SEQUENCE {
  dialCtlPeerCfgId                 INTEGER,
  dialCtlPeerCfgIfType             IANAifType,
  dialCtlPeerCfgLowerIf            InterfaceIndexOrZero,
  dialCtlPeerCfgOriginateAddress   DisplayString,
  dialCtlPeerCfgAnswerAddress      DisplayString,
  dialCtlPeerCfgSubAddress         DisplayString,
  dialCtlPeerCfgClosedUserGroup    DisplayString,
  dialCtlPeerCfgSpeed              INTEGER,
  dialCtlPeerCfgInfoType           INTEGER,
  dialCtlPeerCfgPermission         INTEGER,
  dialCtlPeerCfgInactivityTimer    INTEGER,
  dialCtlPeerCfgMinDuration        INTEGER,
  dialCtlPeerCfgMaxDuration        INTEGER,
  dialCtlPeerCfgCarrierDelay       INTEGER,
  dialCtlPeerCfgCallRetries        INTEGER,
  dialCtlPeerCfgRetryDelay         INTEGER,
  dialCtlPeerCfgFailureDelay       INTEGER,
  dialCtlPeerCfgTrapEnable         INTEGER,
  dialCtlPeerCfgStatus             RowStatus
}

dialCtlPeerCfgId OBJECT-TYPE
SYNTAX      INTEGER (1..2147483647)
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
  "This object identifies a single peer. There may
  be several entries in this table for one peer,
  defining different ways of reaching this peer.
  Thus, there may be several entries in this table
  with the same value of dialCtlPeerCfgId.
  Multiple entries for one peer may be used to support
  multilink as well as backup lines.
  A single peer will be identified by a unique value
  of this object. Several entries for one peer MUST
  have the same value of dialCtlPeerCfgId, but different
  ifEntries and thus different values of ifIndex."
::= { dialCtlPeerCfgTable 1 }

dialCtlPeerCfgIfType OBJECT-TYPE
SYNTAX      IANAifType
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
  "The interface type to be used for calling this peer."
In case of ISDN, the value of isdn(63) is to be used."

DEFVAL { other } ::= { dialCtlPeerCfgEntry 2 }

dialCtlPeerCfgLowerIf OBJECT-TYPE
SYNTAX   InterfaceIndexOrZero
MAX-ACCESS read-create
STATUS   current
DESCRIPTION
"ifIndex value of an interface the peer will have to be
called on. For example, on an ISDN interface, this can be
the ifIndex value of a D channel or the ifIndex value of a
B channel, whatever is appropriate for a given peer.
As an example, for Basic Rate leased lines it will be
necessary to specify a B channel ifIndex, while for
semi-permanent connections the D channel ifIndex has
to be specified.
If the interface can be dynamically assigned, this object
has a value of zero."

DEFVAL { 0 } ::= { dialCtlPeerCfgEntry 3 }


dialCtlPeerCfgOriginateAddress OBJECT-TYPE
SYNTAX   DisplayString
MAX-ACCESS read-create
STATUS   current
DESCRIPTION
"Call Address at which the peer will be called.
Think of this as the set of characters following 'ATDT '
or the 'phone number' included in a D channel call request.
The structure of this information will be switch type
specific. If there is no address information required
for reaching the peer, i.e., for leased lines,
this object will be a zero length string."
 ::= { dialCtlPeerCfgEntry 4 }

dialCtlPeerCfgAnswerAddress OBJECT-TYPE
SYNTAX   DisplayString
MAX-ACCESS read-create
STATUS   current
DESCRIPTION
"Calling Party Number information element, as for example
passed in an ISDN SETUP message by a PBX or switch,
for incoming calls.
This address can be used to identify the peer.
If this address is either unknown or identical
to dialCtlPeerCfgOriginateAddress, this object will be
a zero length string.
DEFVAL { "" }
::= { dialCtlPeerCfgEntry 5 }

dialCtlPeerCfgSubAddress OBJECT-TYPE
SYNTAX DisplayString
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Subaddress at which the peer will be called. If the subaddress is undefined for the given media or unused, this is a zero length string."
DEFVAL { "" }
::= { dialCtlPeerCfgEntry 6 }

dialCtlPeerCfgClosedUserGroup OBJECT-TYPE
SYNTAX DisplayString
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Closed User Group at which the peer will be called. If the Closed User Group is undefined for the given media or unused, this is a zero length string."
REFERENCE
"Q.931, chapter 4.6.1."
DEFVAL { "" }
::= { dialCtlPeerCfgEntry 7 }

dialCtlPeerCfgSpeed OBJECT-TYPE
SYNTAX INTEGER (0..2147483647)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The desired information transfer speed in bits/second when calling this peer. The detailed media specific information, e.g. information type and information transfer rate for ISDN circuits, has to be extracted from this object. If the transfer speed to be used is unknown or the default speed for this type of interfaces, the value of this object may be zero."
DEFVAL { 0 }
::= { dialCtlPeerCfgEntry 8 }

dialCtlPeerCfgInfoType OBJECT-TYPE
SYNTAX INTEGER { other(1), speech(2),
unrestrictedDigital(3),    -- 64k/s data
unrestrictedDigital64k(4), -- with 56k rate adaption
restrictedDigital(5),
audio31(6),             -- 3.1 kHz audio
audio7(7),              -- 7 kHz audio
video(8),
packetSwitched(9),
fax(10)
}
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The Information Transfer Capability to be used when
calling this peer.

speech(2) refers to a non-data connection, whereas
audio31(6) and audio7(7) refer to data mode
connections."
DEFVAL { other }
::= { dialCtlPeerCfgEntry 9 }

dialCtlPeerCfgPermission OBJECT-TYPE
SYNTAX   INTEGER {
    originate(1),
    answer(2),
    both(3), -- both originate & answer
    callback(4),
    none(5)
}
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Applicable permissions. callback(4) either rejects the
call and then calls back, or uses the ‘Reverse charging’
information element if it is available.
Note that callback(4) is supposed to control charging, not
security, and applies to callback prior to accepting a
call. Callback for security reasons can be handled using
PPP callback."
DEFVAL { both }
::= { dialCtlPeerCfgEntry 10 }

dialCtlPeerCfgInactivityTimer OBJECT-TYPE
SYNTAX   INTEGER (0..2147483647)
UNITS    "seconds"
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The connection will be automatically disconnected
if no longer carrying useful data for a time
period, in seconds, specified in this object.
Useful data in this context refers to forwarding
packets, including routing information; it
excludes the encapsulator maintenance frames.
A value of zero means the connection will not be
automatically taken down due to inactivity,
which implies that it is a dedicated circuit."

DEFVAL { 0 }
 ::= { dialCtlPeerCfgEntry 11 }

dialCtlPeerCfgMinDuration OBJECT-TYPE
SYNTAX INTEGER (0..2147483647)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Minimum duration of a call in seconds, starting from the
time the call is connected until the call is disconnected.
This is to accomplish the fact that in most countries
charging applies to units of time, which should be matched
as closely as possible."

DEFVAL { 0 }
 ::= { dialCtlPeerCfgEntry 12 }

dialCtlPeerCfgMaxDuration OBJECT-TYPE
SYNTAX INTEGER (0..2147483647)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Maximum call duration in seconds. Zero means ‘unlimited’."

DEFVAL { 0 }
 ::= { dialCtlPeerCfgEntry 13 }

dialCtlPeerCfgCarrierDelay OBJECT-TYPE
SYNTAX INTEGER (0..2147483647)
UNITS "seconds"
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The call timeout time in seconds. The default value
of zero means that the call timeout as specified for
the media in question will apply."

DEFVAL { 0 }
 ::= { dialCtlPeerCfgEntry 14 }

dialCtlPeerCfgCallRetries OBJECT-TYPE
SYNTAX INTEGER (0..2147483647)
 MAX-ACCESS  read-create
 STATUS    current
 DESCRIPTION
 "The number of calls to a non-responding address
 that may be made. A retry count of zero means
 there is no bound. The intent is to bound
 the number of successive calls to an address
 which is inaccessible, or which refuses those calls.

 Some countries regulate the number of call retries
 to a given peer that can be made."
 DEFVAL    { 0 }
 ::= { dialCtlPeerCfgEntry 15 }

dialCtlPeerCfgRetryDelay OBJECT-TYPE
 SYNTAX    INTEGER (0..2147483647)
 UNITS     "seconds"
 MAX-ACCESS read-create
 STATUS    current
 DESCRIPTION
 "The time in seconds between call retries if a peer
 cannot be reached.
 A value of zero means that call retries may be done
 without any delay."
 DEFVAL    { 0 }
 ::= { dialCtlPeerCfgEntry 16 }

dialCtlPeerCfgFailureDelay OBJECT-TYPE
 SYNTAX    INTEGER (0..2147483647)
 UNITS     "seconds"
 MAX-ACCESS read-create
 STATUS    current
 DESCRIPTION
 "The time in seconds after which call attempts are
 to be placed again after a peer has been noticed
 to be unreachable, i.e. after dialCtlPeerCfgCallRetries
 unsuccessful call attempts.
 A value of zero means that a peer will not be called
 again after dialCtlPeerCfgCallRetries unsuccessful call
 attempts."
 DEFVAL    { 0 }
 ::= { dialCtlPeerCfgEntry 17 }

dialCtlPeerCfgTrapEnable OBJECT-TYPE
 SYNTAX    INTEGER {
             enabled(1),
             disabled(2)
           }

RFC 2128  Dial Control MIB  March 1997

MAX-ACCESS  read-create
STATUS      current
DESCRIPTION  "This object indicates whether dialCtlPeerCallInformation
              and dialCtlPeerCallSetup traps should be generated for
              this peer."
DEFVAL      { disabled }
 ::= { dialCtlPeerCfgEntry 18 }

dialCtlPeerCfgStatus OBJECT-TYPE
SYNTAX      RowStatus
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION  "Status of one row in this table."
 ::= { dialCtlPeerCfgEntry 19 }

-- Peer statistics table

dialCtlPeerStatsTable OBJECT-TYPE
SYNTAX      SEQUENCE OF DialCtlPeerStatsEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION  "Statistics information for each peer entry. There will be one entry in this table for each entry in the dialCtlPeerCfgTable."
 ::= { dialCtlPeer 2 }

dialCtlPeerStatsEntry OBJECT-TYPE
SYNTAX      DialCtlPeerStatsEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION  "Statistics information for a single Peer. This entry is effectively permanent, and contains information describing the last call attempt as well as supplying statistical information."
AUGMENTS      { dialCtlPeerCfgEntry }
 ::= { dialCtlPeerStatsTable 1 }

DialCtlPeerStatsEntry ::= SEQUENCE {
    dialCtlPeerStatsConnectTime          AbsoluteCounter32,
    dialCtlPeerStatsChargedUnits         AbsoluteCounter32,
    dialCtlPeerStatsSuccessCalls         AbsoluteCounter32,
    dialCtlPeerStatsFailCalls            AbsoluteCounter32,
    dialCtlPeerStatsAcceptCalls          AbsoluteCounter32,
    dialCtlPeerStatsClearCalls           AbsoluteCounter32,
dialCtlPeerStatsRefuseCalls AbsoluteCounter32,
dialCtlPeerStatsLastDisconnectCause OCTET STRING,
dialCtlPeerStatsLastDisconnectText DisplayString,
dialCtlPeerStatsLastSetupTime TimeStamp
}

dialCtlPeerStatsConnectTime OBJECT-TYPE
SYNTAX AbsoluteCounter32
UNITS "seconds"
MAX-ACCESS read-only
STATUS current
DESCRIPTION "Accumulated connect time to the peer since system startup.
This is the total connect time, i.e. the connect time
for outgoing calls plus the time for incoming calls."
::= { dialCtlPeerStatsEntry 1 }

dialCtlPeerStatsChargedUnits OBJECT-TYPE
SYNTAX AbsoluteCounter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION "The total number of charging units applying to this
peer since system startup.
Only the charging units applying to the local interface,
i.e. for originated calls or for calls with 'Reverse
charging' being active, will be counted here."
::= { dialCtlPeerStatsEntry 2 }

dialCtlPeerStatsSuccessCalls OBJECT-TYPE
SYNTAX AbsoluteCounter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION "Number of completed calls to this peer."
::= { dialCtlPeerStatsEntry 3 }

dialCtlPeerStatsFailCalls OBJECT-TYPE
SYNTAX AbsoluteCounter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION "Number of failed call attempts to this peer since system
startup."
::= { dialCtlPeerStatsEntry 4 }

dialCtlPeerStatsAcceptCalls OBJECT-TYPE
SYNTAX AbsoluteCounter32
MAX-ACCESS  read-only  
STATUS current  
DESCRIPTION "Number of calls from this peer accepted since system startup."  
 ::=  { dialCtlPeerStatsEntry 5 }

dialCtlPeerStatsRefuseCalls OBJECT-TYPE  
SYNTAX AbsoluteCounter32  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION "Number of calls from this peer refused since system startup."  
 ::=  { dialCtlPeerStatsEntry 6 }

dialCtlPeerStatsLastDisconnectCause OBJECT-TYPE  
SYNTAX OCTET STRING (SIZE (0..4))  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION "The encoded network cause value associated with the last call. This object will be updated whenever a call is started or cleared. The value of this object will depend on the interface type as well as on the protocol and protocol version being used on this interface. Some references for possible cause values are given below."  
REFERENCE  
"- Bellcore SR-NWT-001953, Generic Guidelines for ISDN Terminal Equipment On Basic Access Interfaces, chapter 5.2.5.8.  
- Bellcore SR-NWT-002343, ISDN Primary Rate Interface Generic Guidelines for Customer Premises Equipment, chapter 8.2.5.8.  
- ITU-T Q.931, Appendix I.  
- ITU-T X.25, CAUSE and DIAGNOSTIC field values.  
- German Telekom FTZ 1TR6, chapter 3.2.3.4.4.4."  
 ::=  { dialCtlPeerStatsEntry 7 }

dialCtlPeerStatsLastDisconnectText OBJECT-TYPE  
SYNTAX DisplayString  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION "ASCII text describing the reason for the last call termination."
This object exists because it would be impossible for a management station to store all possible cause values for all types of interfaces. It should be used only if a management station is unable to decode the value of dialCtlPeerStatsLastDisconnectCause.

This object will be updated whenever a call is started or cleared."
::= { dialCtlPeerStatsEntry 8 }
dialCtlPeerStatsLastSetupTime OBJECT-TYPE
SYNTAX      TimeStamp
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The value of sysUpTime when the last call to this peer was started.
For ISDN media, this will be the time when the setup message was received from or sent to the network.
This object will be updated whenever a call is started or cleared."
::= { dialCtlPeerStatsEntry 9 }

--
-- the active call group
--
callActive OBJECT IDENTIFIER ::= { dialControlMibObjects 3 }

-- callActiveTable
-- Table to store active call information.
-- These calls could be circuit switched or they could
-- be virtual circuits.
-- An entry will be created when a call is started and deleted
-- when a call is cleared.
callActiveTable OBJECT-TYPE
SYNTAX      SEQUENCE OF CallActiveEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
"A table containing information about active calls to a specific destination."
::= { callActive 1 }
callActiveEntry OBJECT-TYPE
SYNTAX      CallActiveEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
"The information regarding a single active Connection.
An entry in this table will be created when a call is
started. An entry in this table will be deleted when
an active call clears."
INDEX       { callActiveSetupTime, callActiveIndex }
::= { callActiveTable 1 }

CallActiveEntry ::= 
  SEQUENCE {
    callActiveSetupTime                  TimeStamp,
    callActiveIndex                      INTEGER,
    callActivePeerAddress                DisplayString,
    callActivePeerSubAddress             DisplayString,
    callActivePeerId                     INTEGER,
    callActivePeerIfIndex                INTEGER,
    callActiveLogicalIfIndex             InterfaceIndexOrZero,
    callActiveConnectTime                TimeStamp,
    callActiveCallState                  INTEGER,
    callActiveCallOrigin                 INTEGER,
    callActiveChargedUnits               AbsoluteCounter32,
    callActiveInfoType                   INTEGER,
    callActiveTransmitPackets            AbsoluteCounter32,
    callActiveTransmitBytes              AbsoluteCounter32,
    callActiveReceivePackets             AbsoluteCounter32,
    callActiveReceiveBytes               AbsoluteCounter32
  }

callActiveSetupTime OBJECT-TYPE
SYNTAX      TimeStamp
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
"The value of sysUpTime when the call associated to this
entry was started. This will be useful for an NMS to
retrieve all calls after a specific time. Also, this object
can be useful in finding large delays between the time the
call was started and the time the call was connected.
For ISDN media, this will be the time when the setup
message was received from or sent to the network."
::= { callActiveEntry 1 }

callActiveIndex OBJECT-TYPE
SYNTAX      INTEGER (1..’7fffffff’h)
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
"Small index variable to distinguish calls that start in
the same hundredth of a second."
::= { callActiveEntry 2 }

callActivePeerAddress OBJECT-TYPE
SYNTAX      DisplayString
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The number this call is connected to. If the number is
not available, then it will have a length of zero."
::= { callActiveEntry 3 }

callActivePeerSubAddress OBJECT-TYPE
SYNTAX      DisplayString
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The subaddress this call is connected to. If the subaddress
is undefined or not available, this will be a zero length
string."
::= { callActiveEntry 4 }

callActivePeerId OBJECT-TYPE
SYNTAX      INTEGER (0..2147483647)
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"This is the Id value of the peer table entry
to which this call was made. If a peer table entry
for this call does not exist or is unknown, the value
of this object will be zero."
::= { callActiveEntry 5 }

callActivePeerIfIndex OBJECT-TYPE
SYNTAX      INTEGER (0..2147483647)
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"This is the ifIndex value of the peer table entry
to which this call was made. If a peer table entry
for this call does not exist or is unknown, the value
of this object will be zero."
::= { callActiveEntry 6 }

callActiveLogicalIfIndex OBJECT-TYPE
SYNTAX      InterfaceIndexOrZero
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
   "This is the ifIndex value of the logical interface through
which this call was made. For ISDN media, this would be
the ifIndex of the B channel which was used for this call.
If the ifIndex value is unknown, the value of this object
will be zero."
::= { callActiveEntry 7 }

callActiveConnectTime OBJECT-TYPE
SYNTAX      TimeStamp
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
   "The value of sysUpTime when the call was connected.
If the call is not connected, this object will have a
value of zero."
::= { callActiveEntry 8 }

callActiveCallState OBJECT-TYPE
SYNTAX      INTEGER {  
   unknown(1),
   connecting(2),
   connected(3),
   active(4)  
}  
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
   "The current call state.
unknown(1)   - The call state is unknown.
connecting(2) - A connection attempt (outgoing call)
                is being made.
connected(3)  - An incoming call is in the process
                of validation.
active(4)     - The call is active.
   "
::= { callActiveEntry 9 }

callActiveCallOrigin OBJECT-TYPE
SYNTAX      INTEGER {  
   originate(1),
   answer(2),
   callback(3)  
}  
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The call origin."
::= { callActiveEntry 10 }

callActiveChargedUnits OBJECT-TYPE
SYNTAX  AbsoluteCounter32
MAX-ACCESS read-only
STATUS   current
DESCRIPTION
"The number of charged units for this connection.
For incoming calls or if charging information is
not supplied by the switch, the value of this object
will be zero."
::= { callActiveEntry 11 }

callActiveInfoType OBJECT-TYPE
SYNTAX   INTEGER {
        other(1),       -- e.g. for non-isdn media
        speech(2),
        unrestrictedDigital(3),   -- 64k/s data
        unrestrictedDigital156(4), -- with 56k rate adaption
        restrictedDigital(5),
        audio31(6),       -- 3.1 kHz audio
        audio7(7),        -- 7 kHz audio
        video(8),
        packetSwitched(9),
        fax(10)
    }
MAX-ACCESS   read-only
STATUS       current
DESCRIPTION
"The information type for this call."
::= { callActiveEntry 12 }

callActiveTransmitPackets OBJECT-TYPE
SYNTAX  AbsoluteCounter32
MAX-ACCESS read-only
STATUS   current
DESCRIPTION
"The number of packets which were transmitted for this
call."
::= { callActiveEntry 13 }

callActiveTransmitBytes OBJECT-TYPE
SYNTAX  AbsoluteCounter32
MAX-ACCESS read-only
STATUS   current
DESCRIPTION
"The number of bytes which were transmitted for this call."
::= { callActiveEntry 14 }

callActiveReceivePackets OBJECT-TYPE
SYNTAX AbsoluteCounter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION "The number of packets which were received for this call."
::= { callActiveEntry 15 }

callActiveReceiveBytes OBJECT-TYPE
SYNTAX AbsoluteCounter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION "The number of bytes which were received for this call."
::= { callActiveEntry 16 }

--
-- the call history group
--

callHistory OBJECT IDENTIFIER ::= { dialControlMibObjects 4 }

callHistoryTableMaxLength OBJECT-TYPE
SYNTAX INTEGER (0..2147483647)
MAX-ACCESS read-write
STATUS current
DESCRIPTION "The upper limit on the number of entries that the callHistoryTable may contain. A value of 0 will prevent any history from being retained. When this table is full, the oldest entry will be deleted and the new one will be created."
::= { callHistory 1 }

callHistoryRetainTimer OBJECT-TYPE
SYNTAX INTEGER (0..2147483647)
UNITS "minutes"
MAX-ACCESS read-write
STATUS current
DESCRIPTION "The minimum amount of time that an callHistoryEntry will be maintained before being deleted. A value of 0 will prevent any history from being retained in the
callHistoryTable, but will neither prevent callCompletion
traps being generated nor affect other tables.
::= { callHistory 2 }

-- callHistoryTable
-- Table to store the past call information. The Destination number
-- and the call connect and disconnect time, the disconnection cause
-- are stored. These calls could be circuit switched or they could
-- be virtual circuits. History of each and every call is stored,
-- of successful calls as well as of unsuccessful and rejected calls.
-- An entry will be created when a call is cleared.

callHistoryTable OBJECT-TYPE
SYNTAX SEQUENCE OF CallHistoryEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"A table containing information about specific
calls to a specific destination."
::= { callHistoryTable 1 }

CallHistoryEntry ::= SEQUENCE {
callHistoryPeerAddress               DisplayString,
callHistoryPeerSubAddress            DisplayString,
callHistoryPeerId                    INTEGER,
callHistoryPeerIfIndex               INTEGER,
callHistoryLogicalIfIndex            InterfaceIndex,
callHistoryDisconnectCause           OCTET STRING,
callHistoryDisconnectText            DisplayString,
callHistoryConnectTime               TimeStamp,
callHistoryDisconnectTime            TimeStamp,
callHistoryCallOrigin                INTEGER,
callHistoryChargedUnits              AbsoluteCounter32,
callHistoryInfoType                  INTEGER,
callHistoryTransmitPackets           AbsoluteCounter32,
callHistoryTransmitBytes             AbsoluteCounter32,
callHistoryReceivePackets            AbsoluteCounter32,
callHistoryReceiveBytes AbsoluteCounter32

callHistoryPeerAddress OBJECT-TYPE
SYNTAX     DisplayString
MAX-ACCESS read-only
STATUS     current
DESCRIPTION
    "The number this call was connected to. If the number is
    not available, then it will have a length of zero."
 ::= { callHistoryEntry 1 }

callHistoryPeerSubAddress OBJECT-TYPE
SYNTAX     DisplayString
MAX-ACCESS read-only
STATUS     current
DESCRIPTION
    "The subaddress this call was connected to. If the subaddress
    is undefined or not available, this will be a zero length
    string."
 ::= { callHistoryEntry 2 }

callHistoryPeerId OBJECT-TYPE
SYNTAX     INTEGER (0..2147483647)
MAX-ACCESS read-only
STATUS     current
DESCRIPTION
    "This is the Id value of the peer table entry
to which this call was made. If a peer table entry
for this call does not exist, the value of this object
will be zero."
 ::= { callHistoryEntry 3 }

callHistoryPeerIfIndex OBJECT-TYPE
SYNTAX     INTEGER (0..2147483647)
MAX-ACCESS read-only
STATUS     current
DESCRIPTION
    "This is the ifIndex value of the peer table entry
to which this call was made. If a peer table entry
for this call does not exist, the value of this object
will be zero."
 ::= { callHistoryEntry 4 }

callHistoryLogicalIfIndex OBJECT-TYPE
SYNTAX     InterfaceIndex
MAX-ACCESS read-only
STATUS     current
DESCRIPTION
"This is the ifIndex value of the logical interface through which this call was made. For ISDN media, this would be the ifIndex of the B channel which was used for this call."

::= { callHistoryEntry 5 }

callHistoryDisconnectCause OBJECT-TYPE
SYNTAX OCTET STRING (SIZE (0..4))
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The encoded network cause value associated with this call. The value of this object will depend on the interface type as well as on the protocol and protocol version being used on this interface. Some references for possible cause values are given below."

REFERENCE
- Bellcore SR-NWT-001953, Generic Guidelines for ISDN Terminal Equipment On Basic Access Interfaces, chapter 5.2.5.8.
- Bellcore SR-NWT-002343, ISDN Primary Rate Interface Generic Guidelines for Customer Premises Equipment, chapter 8.2.5.8.
- ITU-T Q.931, Appendix I.
- ITU-T X.25, CAUSE and DIAGNOSTIC field values.
- German Telekom FTZ 1TR6, chapter 3.2.3.4.4.4."

::= { callHistoryEntry 6 }

callHistoryDisconnectText OBJECT-TYPE
SYNTAX DisplayString
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"ASCII text describing the reason for call termination. This object exists because it would be impossible for a management station to store all possible cause values for all types of interfaces. It should be used only if a management station is unable to decode the value of dialCtlPeerStatsLastDisconnectCause."

::= { callHistoryEntry 7 }

callHistoryConnectTime OBJECT-TYPE
SYNTAX TimeStamp
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The value of sysUpTime when the call was connected."
::= { callHistoryEntry 8 }

callHistoryDisconnectTime OBJECT-TYPE
SYNTAX      TimeStamp
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The value of sysUpTime when the call was disconnected."
::= { callHistoryEntry 9 }

callHistoryCallOrigin OBJECT-TYPE
SYNTAX      INTEGER {
            originate(1),
            answer(2),
            callback(3)
        }
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The call origin."
::= { callHistoryEntry 10 }

callHistoryChargedUnits OBJECT-TYPE
SYNTAX      AbsoluteCounter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The number of charged units for this connection. For incoming calls or if charging information is not supplied by the switch, the value of this object will be zero."
::= { callHistoryEntry 11 }

callHistoryInfoType OBJECT-TYPE
SYNTAX      INTEGER {
            other(1),           -- e.g. for non-isdn media
            speech(2),          -- 64k/s data
            unrestrictedDigital(3),  -- 64k/s data
            unrestrictedDigital156(4),  -- with 56k rate adaption
            restrictedDigital(5),
            audio31(6),         -- 3.1 kHz audio
            audio7(7),          -- 7 kHz audio
            video(8),
            packetSwitched(9),
            fax(10)
        }
MAX-ACCESS  read-only
STATUS current
DESCRIPTION  "The information type for this call."
 ::= { callHistoryEntry 12 }

callHistoryTransmitPackets OBJECT-TYPE
SYNTAX AbsoluteCounter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION  "The number of packets which were transmitted while this
call was active."
 ::= { callHistoryEntry 13 }

callHistoryTransmitBytes OBJECT-TYPE
SYNTAX AbsoluteCounter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION  "The number of bytes which were transmitted while this
call was active."
 ::= { callHistoryEntry 14 }

callHistoryReceivePackets OBJECT-TYPE
SYNTAX AbsoluteCounter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION  "The number of packets which were received while this
call was active."
 ::= { callHistoryEntry 15 }

callHistoryReceiveBytes OBJECT-TYPE
SYNTAX AbsoluteCounter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION  "The number of bytes which were received while this
call was active."
 ::= { callHistoryEntry 16 }

-- Traps related to Connection management

dialControlMibTrapPrefix OBJECT IDENTIFIER ::= { dialControlMib 2 }
dialControlMibTraps OBJECT IDENTIFIER ::= { dialControlMibTrapPrefix 0 }

dialCtlPeerCallInformation NOTIFICATION-TYPE
OBJECTS {
callHistoryPeerId,
callHistoryPeerIfIndex,
callHistoryLogicalIfIndex,
ifOperStatus,
callHistoryPeerAddress,
callHistoryPeerSubAddress,
callHistoryDisconnectCause,
callHistoryConnectTime,
callHistoryDisconnectTime,
callHistoryInfoType,
callHistoryCallOrigin

} STATUS current

DESCRIPTION
"This trap/inform is sent to the manager whenever a successful call clears, or a failed call attempt is determined to have ultimately failed. In the event that call retry is active, then this is after all retry attempts have failed. However, only one such trap is sent in between successful call attempts; subsequent call attempts result in no trap. ifOperStatus will return the operational status of the virtual interface associated with the peer to whom this call was made to."

::= { dialControlMibTraps 1 }

dialCtlPeerCallSetup NOTIFICATION-TYPE

OBJECTS {
    callActivePeerId,
callActivePeerIfIndex,
callActiveLogicalIfIndex,
ifOperStatus,
callActivePeerAddress,
callActivePeerSubAddress,
callActiveInfoType,
callActiveCallOrigin

} STATUS current

DESCRIPTION
"This trap/inform is sent to the manager whenever a call setup message is received or sent. ifOperStatus will return the operational status of the virtual interface associated with the peer to whom this call was made to."

::= { dialControlMibTraps 2 }

-- conformance information
RFC 2128

Dial Control MIB

March 1997

dialControlMibConformance OBJECT IDENTIFIER ::= 
    { dialControlMib 3 }
dialControlMibCompliances OBJECT IDENTIFIER ::= 
    { dialControlMibConformance 1 }
dialControlMibGroups OBJECT IDENTIFIER ::= 
    { dialControlMibConformance 2 }

-- compliance statements

dialControlMibCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
        "The compliance statement for entities which
         implement the DIAL CONTROL MIB"
    MODULE -- this module
    MANDATORY-GROUPS
        { dialControlGroup, callActiveGroup, callHistoryGroup,
          callNotificationsGroup }
    ::= { dialControlMibCompliances 1 }

-- units of conformance

dialControlGroup OBJECT-GROUP
    OBJECTS {
        dialCtlAcceptMode,
        dialCtlTrapEnable,
        dialCtlPeerCfgIfType,
        dialCtlPeerCfgLowerIf,
        dialCtlPeerCfgOriginateAddress,
        dialCtlPeerCfgAnswerAddress,
        dialCtlPeerCfgSubAddress,
        dialCtlPeerCfgClosedUserGroup,
        dialCtlPeerCfgSpeed,
        dialCtlPeerCfgInfoType,
        dialCtlPeerCfgPermission,
        dialCtlPeerCfgInactivityTimer,
        dialCtlPeerCfgMinDuration,
        dialCtlPeerCfgMaxDuration,
        dialCtlPeerCfgCarrierDelay,
        dialCtlPeerCfgCallRetries,
        dialCtlPeerCfgRetryDelay,
        dialCtlPeerCfgFailureDelay,
        dialCtlPeerCfgTrapEnable,
        dialCtlPeerCfgStatus,
        dialCtlPeerStatsConnectTime,
        dialCtlPeerStatsChargedUnits,
        dialCtlPeerStatsSuccessCalls,
        dialCtlPeerStatsFailCalls,
dialCtlPeerStatsAcceptCalls,
dialCtlPeerStatsRefuseCalls,
dialCtlPeerStatsLastDisconnectCause,
dialCtlPeerStatsLastDisconnectText,
dialCtlPeerStatsLastSetupTime
}
STATUS current
DESCRIPTION
"A collection of objects providing the DIAL CONTROL
 capability."
 ::= { dialControlMibGroups 1 }

callActiveGroup OBJECT-GROUP
  OBJECTS {
    callActivePeerAddress,
callActivePeerSubAddress,
callActivePeerId,
callActivePeerIfIndex,
callActiveLogicalIfIndex,
callActiveConnectTime,
callActiveCallState,
callActiveCallOrigin,
callActiveChargedUnits,
callActiveInfoType,
callActiveTransmitPackets,
callActiveTransmitBytes,
callActiveReceivePackets,
callActiveReceiveBytes
  }
  STATUS current
  DESCRIPTION
  "A collection of objects providing the active call
   capability."
  ::= { dialControlMibGroups 2 }

callHistoryGroup OBJECT-GROUP
  OBJECTS {
    callHistoryTableMaxLength,
callHistoryRetainTimer,
callHistoryPeerAddress,
callHistoryPeerSubAddress,
callHistoryPeerId,
callHistoryPeerIfIndex,
callHistoryLogicalIfIndex,
callHistoryDisconnectCause,
callHistoryDisconnectText,
callHistoryConnectTime,
callHistoryConnectTime,
}
callHistoryCallOrigin,
callHistoryChargedUnits,
callHistoryInfoType,
callHistoryTransmitPackets,
callHistoryTransmitBytes,
callHistoryReceivePackets,
callHistoryReceiveBytes
)
STATUS current
DESCRIPTION "A collection of objects providing the Call History
capability."
::= { dialControlMibGroups 3 }
callNotificationsGroup NOTIFICATION-GROUP
  NOTIFICATIONS { dialCtlPeerCallInformation, dialCtlPeerCallSetup }
  STATUS current
  DESCRIPTION "The notifications which a Dial Control MIB entity is
  required to implement."
::= { dialControlMibGroups 4 }
END

4. Acknowledgments

This document was produced by the ISDN MIB Working Group. Special
thanks is due to the following persons:

Ed Alcoff
Fred Baker
Bibek A. Das
Ken Grigg
Jeffrey T. Johnson
Glenn Kime
Oliver Korfmacher
Kedar Madineni
Bill Miskovetz
David M. Piscitello
Lisa A. Phifer
Randy Roberts
Hascall H. Sharp
Hongchi Shih
Robert Snyder
Bob Stewart
Ron Stoughton
James Watt
5. References


6. Security Considerations

Information in this MIB may be used by upper protocol layers for security purpose.

The implementor should be aware that supporting generic peers as described in section 3.4 may cause a security hole. The user would not know where a call is from, which could potentially allow unauthorized access if there is no other authentication scheme, e.g. PPP authentication, available.
7. Author’s Address

Guenter Roeck
Cisco Systems
170 West Tasman Drive
San Jose, CA 95134
U.S.A.

Phone: +1 408 527 3143
EMail: groeck@cisco.com