

Definitions of Managed Objects for Data Link Switching  
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 specification defines an extension to the Management Information Base (MIB) for use with SNMP-based network management. In particular, it defines objects for configuring, monitoring, and controlling Data Link Switches (DLSw) [1].

This memo specifies a MIB module in a manner that is both compliant to the SNMPv2 SMI [2], and semantically identical to the SNMPv1 definitions [3].

Table of Contents

1.0	The SNMPv2 Network Management Framework	2
1.1	Object Definitions	2
2.0	Overview	2
2.1	Relation to Interface Group (RFC 1573) [8]	2
2.2	Relation to Underlying DLC Layer	3
2.3	Relation to SDLC MIB (RFC 1747)	3
2.4	DLSw MIB Structure	4
2.4.1	Compliance	4
2.5	DLSw MIB Usage	5
2.5.1	Cooperative DLSw nodes	5
2.5.2	Setting capabilities exchange-related objects	5
2.5.3	Examples of Tasks Using This MIB	6
3.0	Definitions	11
4.0	Acknowledgements	89
5.0	References	89
6.0	Security Considerations	90

7.0 Authors' Addresses . . . . . 90

1.0 The SNMPv2 Network Management Framework

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

RFC 1902 [2] which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management.

STD 17, RFC 1213 [4] defines MIB-II, the core set of managed objects for the Internet suite of protocols.

STD 15, RFC 1157 [5] and RFC 1905 [6] which define two versions of the protocol used for network access to 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.

2.0 Overview

This memo identifies the set of objects for configuring, monitoring, and controlling Data Link Switches.

2.1 Relation to Interface Group (RFC 1573) [8]

- o ifIndex is used as the index into dlswIfTable, which shows and controls the interfaces that DLSw is active on.
- o Local entries in the MAC address and NetBIOS (NB) name caches can point to an ifEntry to indicate the interface through which DLSw can reach that MAC address or NB name. See the objects dlswDirMacLocation and dlswDirNBLocation.
- o Local entries in the circuit table use ifIndex to indicate the interface through which DLSw is connected to the local end station.

See the object `dlswCircuitS1Index`.

- o `ifIndex` is the primary index into `dlswSdlcLsTable`, which lists the SDLC stations DLSw is serving.

## 2.2 Relation to Underlying DLC Layer

The DLSw MIB does not duplicate the information in the MIBs for the DLC layer underneath it. Instead, each circuit table entry contains a pointer to a conceptual row in an underlying enterprise-specific or standard DLC MIB.

Using the 802.2 LLC management as an example, the following rules should be considered when developing new DLSw related DLC MIBs, and when implementing the interactions between DLSw MIB and DLC MIBs:

- o The referenced row should represent the local LLC-2 (and/or LLC-1, if supported) link station that DLSw is using. In the current 802.2 LLC MIB draft, this might be a row of one of the tables `llcCcAdminTable`, `llcCcOperTable`, or `llcCcStatsTable`.

A circuit using local LLC services will therefore have `dlswCircuitS1DlcType = llc`, and `dlswCircuitS1Dlc` = pointer to an LLC MIB table row.

- o Because DLSw is the user of LLC services, it is generally preferable to initiate administrative actions using the DLSw MIB and allow DLSw to control LLC directly, rather than starting with LLC MIB administrative actions. For example, a hung circuit should be disconnected by setting `dlswCircuitState`, as opposed to setting `llcCcAdminStatus` to disable the LLC part of the circuit. Similarly, setting bits in `dlswIfSapList` will cause row creation in `llcSapOperTable` as well as set the necessary DLSw-LLC relationship.

## 2.3 Relation to SDLC MIB (RFC 1747)

The general comments stated in 2.2, "Relation to Underlying DLC Layer" apply to the SDLC MIB. The following apply if the DLSw MIB is implemented in a product that also implements RFC 1747 [9]:

- o The row referenced from `dlswCircuitS1Dlc` should represent the local SDLC link station that DLSw is using. This might be a row of one of the tables `sdlcLSAdminTable`, `sdlcLSOperTable`, or `sdlcLSStatsTable`.

A circuit using local SDLC services will therefore have `dlswCircuitS1DlcType = sdlc`, and `dlswCircuitS1Dlc` = OID of one of these table rows.

- o dlswSdlcLsTable uses the same indices that are used to index link station information in RFC 1747. This table provides a mapping between this native SDLC addressing (interface, link station address) and the addressing used in the DLSw domain (local MAC and SAP).

## 2.4 DLSw MIB Structure

See 3 .0, "Definitions" on page 11 for a diagram outlining the DLSw MIB structure. The following groups of objects are included:

dlswNode	Objects related to this DLSw node's configuration, monitoring and control.
dlswTConn	Objects relating to transport connections to this DLSw's partner nodes.
dlswInterface	Objects configured for this DLSw relating to its local interfaces.
dlswDirectory	Objects reflecting this DLSw's view of where end-station resources (MAC addresses and NetBIOS names) are located.
dlswCircuit	Objects showing the end-station connections that DLSw currently has established, or that are coming up or have gone down.
dlswSDLC	Objects configured for this DLSw's SDLC-attached end stations.

### 2.4.1 Compliance

The MIB provides the following compliance statements:

dlswCoreCompliance	Defines the minimum support required of all implementations. Note that for this and the other compliance statements, NetBIOS-related objects are grouped separately because the DLSw Version 1 Standard [1] does not require NetBIOS support.
dlswTConnTCPCompliance	Defines the minimum support required of implementations that use TCP as a transport protocol.
dlswDirCompliance	Defines the minimum support required of implementations that support some sort of

directory function.

`dlswDirLocateCompliance` Defines the minimum support required of implementations that support a directory function and also support the ordered retrieval of the entries that match a given resource.

`dlswSdlcCompliance` Defines the minimum support required of implementations that support SDLC-attached end stations.

## 2.5 DLSw MIB Usage

### 2.5.1 Cooperative DLSw nodes

To reduce the size of the MIB, thus the amount of data that each agent needs to keep, the information that usually could be made available in two partner nodes (e.g., information exchanged between them) is only defined in the MIB as the info received. That is, there are no objects defined for the info sent. In order to form the complete picture of the state of a resource, the manager needs to retrieve info from multiple DLSw nodes. An example is that the SAP list, NETBIOS list and MAC list are kept at the receiving end of a DLSw capabilities exchange (the sender does not save what it sent to each partner).

Note well: The DLSw protocol does not specify a technique for a manager to correlate the transport address of the partner managed DLSw node and the transport address that the management protocol uses.

### 2.5.2 Setting capabilities exchange-related objects

This MIB supports changes to DLSw variables whose change should be reported to DLSw partner nodes in a "run-time" capabilities exchange. Since a DLSw node normally unicasts these capabilities messages to all its active partners, frequent changes to these variables can result in excessive network traffic. To avoid this problem, developers of network management applications using this MIB should try to group all such changes in a few SNMP SET requests, and should send them in bulk. Agent developers should implement a technique to group a number of changes into a single capabilities exchange message. One possible approach is to send a run-time capabilities message only if no capabilities-related changes have been received for a pre-defined period of time.

### 2.5.3 Examples of Tasks Using This MIB

#### 2.5.3.1 Configuring DLSw to actively connect to a specific TCP/IP partner

Create a conceptual row in `dlswTConnConfigTable` with: `Index` = the highest the managed station has used so far + 1; `TDomain` = `dlswTCPDomain`; `LocalTAddr` = this node's DLSw IP address; `RemoteTAddr` = the partner's DLSw IP address; `EntryType` = `individual`; `SetupType` = `activePersistent`. Note that determining the index to use may require dumping the `TConnConfigTable`, but this will not typically be a large table. If the DLSw node rejects the row creation due to index collision, the management station should increment its index value and try again.

#### 2.5.3.2 Configuring DLSw to passively accept any partner

Create a conceptual row in `dlswTConnConfigTable` as above but with: `RemoteTAddr` = 0; `EntryType` = `global`; `SetUpType` = `passive`. Every individual transport connection accepted as a result of this global row will inherit the configuration values from this row.

To prevent a specific remote node from being passively accepted as a partner, create another row with: `RemoteTAddr` = that node's IP address; `EntryType` = `individual`; `SetupType` = `excluded`.

#### 2.5.3.3 Configuring DLSw to allow or connect to a group of partners

Define a conceptual row in `dlswTConnConfigTable` as above but with: `EntryType` = `group`; `GroupDefinition` = pointer to an enterprise-specific representation of a group. For example, a group definition might consist of an IP address value and mask, or a multicast IP address. Every individual transport connection accepted as a result of this group row will inherit the configuration values from this row.

When a group is created that has some overlap with entries where `EntryType` = `individual` (there will always be this overlap when a global row exists), the DLSw node must use the configured rows using a "most specific match wins" rule. That is, the entry in `TConnConfigTable` with the remote address most nearly matching an incoming connection should be used to provide the values for the new connection. For equal matches, the choice of `TConnConfigTable` entry is up to the DLSw node implementation. Note that the management station should never create two `TConnConfig` rows with duplicate remote addressing values.

#### 2.5.3.4 Identifying the protocol level of a partner DLSw

If the partner DLSw has implemented at least the AIW Version 1 DLSw Standard [1], the AIW version and release number for the DLSw protocol is accessible from `dlswTConnOperPartnerVersion`. If `TConnOperPartnerVersion` is a string of zero length but the `TConnOperState = 'connected'` state (i.e., is not still performing capabilities exchange), the partner DLSw can be assumed to be an RFC 1434+ node.

#### 2.5.3.5 Recycling a transport connection

Quiesce or forcibly disconnect the transport connection by setting `TConnOperState` to `'quiescing'` or `'disconnecting'`, and monitor until it moves to the `'disconnected'` state or the `TConnOper` row disappears. The row may disappear because implementations are not required to maintain transport connection information after a transport connection has gone down.

The action required to re-activate the transport connection depends on the value of `TConnConfigSetupType` for the relevant `TConnConfig` row. `ActivePersistent` connections will attempt to come back automatically. `Passive` connections must be re-established from the remote partner. `ActiveOnDemand` connections will be re-established by this node, but only after some end-station operation triggers a circuit setup attempt.

#### 2.5.3.6 Investigating why a transport connection went down

`TConnOperDiscTime` and `TConnOperDiscReason` provide the vital information of the time and the cause of the disconnection of a transport connection and `TConnOperDiscActiveCir` indicates whether end users may have been affected. This MIB does not specify the duration that an agent must make this information available after the disconnection of a transport connection occurs. Manager should try the agent of the partner DLSw, if such information is not available in one DLSw node. Additional information might come from the MIB for the transport protocol (e.g., TCP or LLC). `dlswTConnStat*` and `dlswTConnConfigOpens` give a more general picture of transport connection activity, but can't give specific reasons for problems.

#### 2.5.3.7 Changing the configuration of an active transport connection

Follow this sequence of management protocol set operations:

1. Use `TConnOperConfigIndex` to locate the `TConnConfig` entry that governs the configuration of the transport connection.

2. Change the rowStatus of that conceptual row to notInService. This prevents the transport connection from being connected automatically if TConnConfigSetupType = activePersistent.
3. Quiesce or forcibly disconnect the transport connection by setting TConnOperState to 'quiescing' or 'disconnecting', and monitor until it moves to the 'disconnected' state or the TConnOper row disappears.
4. Change the values of TConnConfig variables as desired.
5. Change the rowStatus of the TConnConfig conceptual row to active. TConnConfigSetupType will subsequently control whether this node will actively seek to re-establish the transport connection, or will wait.

#### 2.5.3.8 Checking configuration validity for an active transport connection

Use TConnOperConfigIndex to identify the row of TConnConfig for the transport connection. If TConnConfigLastModifyTime is greater than TConnOperConnectTime, then one or more of the variables in the TConnConfig row may not be valid for the current state of the active transport connection. This is an exception condition and will not normally be the case.

#### 2.5.3.9 Configuring the interfaces and SAPs DLSw will use

To add DLSw end-station support (not transport connection support) to an interface, create a conceptual row for that ifIndex in the dlswIfTable. For many products, you will specify the same single virtual segment number for all interfaces. Indicate the list of SAPs to be supported by that interface - this could be all 0xFFs if the product has some automatic SAP opening function.

To open or close a SAP to DLSw on an existing interface, simply set or reset the appropriate bit in dlswIfSapList in the table row for that interface.

#### 2.5.3.10 Configuring static MAC address (or NetBIOS name) cache entries

It is common to configure a few static directory entries to preload in the caches of the DLSw nodes and reduce the need for broadcast searches. The following example adds entries to the MAC cache to indicate that a specific MAC address is reachable through two different remote partners:

1. The manager retrieves dlswDirMacCacheNextIndex to get an index assignment from the DLSw node. The DLSw node ensures that the retrieved index will not be reused.

2. The manager creates a conceptual row in `dlswDirMacTable` with:  
Index = the retrieved index; Mac = the MAC address; Mask = all 0xFF's; EntryType = `userConfiguredPublic`; LocationType = `remote`; Location = OID for `dlswTConnConfigEntry` of the 1st partner; Status = `unknown` (recommended for new entries).
3. The manager repeats the preceding 2 steps and creates a second row using Index = second index retrieved; Location = OID for `dlswTConnConfigEntry` of the 2nd partner.

Note that the DLSw node is not obligated to use newly created directory entries in the order in which they were created. It is recommended that entries be used in most-specific match first order, i.e., an entry with a Mask of all 0xFFs should take precedence over one with a "partial wildcard". The relative order of static versus dynamic entries and of "equal length" matches is up to the DLSw implementation.

The `dlswDirStat` objects can be used to get an idea of the success rate for a particular static caching scheme.

#### 2.5.3.11 Seeing where the directory indicates a given resource is

To retrieve all directory information related to a given resource (in this example, a NetBIOS name), the management station should:

1. Retrieve `dlswDirLocateNBLocation` in the `dlswDirLocateNBTable` entry where `NBName` = the fully-specified NetBIOS name without wildcards; `NBMatch` = 1.
2. Use the returned value (i.e., OID) to retrieve the contents of the `dlswDirNBEntry` itself.
3. Repeat the previous two steps with `NBMatch` = 2, 3, ..., until the end of `dlswDirLocateNBTable` is reached.

The DLSw node conveys the precedence relationship of the different matching directory entries by the order in which it returns their OIDs.

#### 2.5.3.12 Investigating circuit bringup failure

Circuit bringup takes place in two stages: explorer flows to locate the target resource (MAC address or NetBIOS name); and establishing the circuit itself. To determine the success of explorer flows, have the origin end station initiate a link establishment to the target, and look later for cache entries for the target MAC address or NetBIOS name. The `dlswTConn*ex*` counters also give some visibility to which transport connections are being used to look for resources. Once circuit establishment is started, an entry of `dlswCircuitTable` for the two MAC/SAP addresses involved is created.

`dlswCircuitEntryTime`, `StateTime`, and `State` may provide useful information about intermediate states the circuit is reaching before becoming disconnected again.

#### 2.5.3.13 Investigating the failure of an established circuit

The variables `dlswCircuitDiscReason*` in the `dlswCircuitTable` provide the key information of the cause of the disconnection of circuits. In addition, the underlying DLC MIBs may provide information at the link station level, and some clues (e.g., DISC or FRMR counters) at the SAP or interface level.

#### 2.5.3.14 Seeing circuit-level traffic statistics

Locate the relevant `dlswCircuitEntry` and follow `dlswCircuitS1Dlc` to a link station-level table entry in the underlying DLC MIB. Move to the corresponding link station's statistics table in the DLC MIB to get counters of frames, bytes, etc. for this circuit.

#### 2.5.3.15 Cutting down the flow of DLSw-related traps

Set some or all of the `dlswTrapCntl*` objects to the value of 'disabled' or 'partial'.

3.0 Definitions

```

-- *****
--
-- The structure of the DLSw MIB (t: indicates table):
--   DLSw MIB
--     |-- Node Group
--     |   |-- Node Identity
--     |   |-- Node Operational Related
--     |   |-- Node Resource
--     |-- Transport Connection Group
--     |   |-- Statistics
--     |   |t- Transport Connection Configuration
--     |   |t- Transport Connection Operation
--     |       |-- capabilities
--     |       |-- Supported SAP List
--     |       |-- statistics
--     |           |-- transport connection itself
--     |           |-- traffic over the transport connection
--     |           |-- directory search activities
--     |           |-- search filtered statistics
--     |           |-- circuits over the transport connection
--     |   |-- Transport Specific
--     |       |-- Tcp
--     |           |t- Transport Connection Config (Tcp Specific)
--     |           |t- Transport Connection Operation (Tcp Specific)
--     |-- Interface Group
--     |   |t- interfaces that DLSw is active on.
--     |-- Directory Group
--     |   |-- Statistics
--     |   |-- Directory Cache
--     |       |t- Directory of MAC addresses
--     |       |t- Directory of NETBIOS names
--     |   |-- Locate
--     |       |t- Directory of Locate MAC
--     |       |t- Directory of Locate NETBIOS
--     |-- Circuit Group
--     |   |-- Statistics
--     |   |t- Circuits
--     |-- Virtual and non-LAN end stations
--     |   |t- SDLC end station
-- *****

```

```
-- *****
-- This MIB module contains objects necessary for management of Data
-- Link Switches.
--
-- Terminology:
-- (1) DLSw:
--     A device which provides data link switching function.
--     Sometimes it is referred as a DLSw or DLSw node.
--     Local DLSw:   The DLSw that the DLSw SNMP Agent is running on.
--     Partner DLSw (or DLSw partner): A DLSw node that is "transport
--     connected" with the local DLSw.  Sometimes the term "DLSw
--     partners" is used to indicate the two ends of a transport
--     connection.
--
-- (2) TCP Connection:
--     Full-duplex (-capable) association defined by a pair of
--     (IP address, port) pairs, running the TCP protocol.  The port
--     addresses in RFC 1795 define two TCP connections between
--     a pair of DLSw nodes, each being used to send data in a
--     single direction.
--     Local:       This end of TCP connection
--     Foreign:     Remote end of TCP connection
--
-- (3) Transport Connection:
--     It is a generic term for a full-duplex reliable connection
--     between DLSw nodes.  This term is used to refer to the
--     association between DLSw nodes without being concerned
--     about whether TCP is the protocol or whether there are
--     one or two TCP connection.
--     (Note: for two TCP connections, the transport connection is
--     opened if and only if both TCP connections are operational.
--     Also note: sometimes race conditions will occur, but the
--     condition should only be temporary.)
--
-- (4) Data Link:
--     An instance of OSI layer-2 procedures for exchanging information
--     using either connection-oriented (e.g., LLC-2) or connectionless
--     (e.g., LLC-1) services.  A DLSw node or pair of partner nodes
--     switches data traffic from stations of one data link to
--     stations of another data link.  Data link switching is
--     transparent to end stations.
--     Source: the end station which sends a message.
--     Destination: the end station which receives a message.
--     (This DLSw role is with respect to a give message)
--
-- (5) Circuit:
--     End-to-end association of two DLC entities through one or
--     two DLSw nodes.  A circuit is the concatenation of two
```

```

--      "data links", optionally with an intervening transport
--      connection.
--      Origin:   the end station which initiates the circuit.
--      Target:   the end station which receives the initiation.
--
-- (6) Link Station:
--      It is one end of an LLC-2 connection. It performs error
--      recovery procedure, retries, and various timers.
--      DLSw terminates LLC-2 connection at each end of DLSw nodes,
--      thus, keepAlive and error recovery on LLC-2 connections are
--      kept to each side of LAN and do not flow through the WAN.
--      A link station is substantiated when SABME is sent/received.
--      All link stations have circuits, but not all circuits
--      have link stations.
--
-- Key assumptions are:
-- (1) The MIB is designed to manage a single DLSw entity.
--
-- (2) A DLSw may support various types of transport connections.
--      - This DLSw MIB module does not restrict the possibility to
--        have, at any given moment, more than one "transport
--        connection" defined or active between two DLSw's.
--      - However, current DLSw architecture does not provide a mechanism,
--        e.g., DLSw host name, to prevent two transport connections of
--        different types between the same two DLSw's.
--
-- (3) This MIB assumes that interface MIB is implemented.  ifIndex
--      is used in this MIB module.
--
-- (4) This MIB assumes that the SDLC MIB (or an equivalent enterprise
--      specific MIB) is implemented, since SDLC-specific objects
--      are not duplicated here.
--
-- (5) This MIB assumes that the LLC-2 MIB (or an equivalent enterprise
--      specific MIB) is implemented, since LLC-related objects are not
--      duplicated here.
--
-- (6) All MACs, SAPs, Ring numbers, ... are in non-canonical form.
--      That is, the most significant bit will be transmitted first.
--
-- *****

```

DLSW-MIB DEFINITIONS ::= BEGIN

IMPORTS

```

    DisplayString, RowStatus,
    RowPointer, TruthValue,
    TEXTUAL-CONVENTION          FROM SNMPv2-TC

```

```

Counter32, Gauge32, TimeTicks,
OBJECT-TYPE, MODULE-IDENTITY,
NOTIFICATION-TYPE           FROM SNMPv2-SMI
MODULE-COMPLIANCE, OBJECT-GROUP,
NOTIFICATION-GROUP         FROM SNMPv2-CONF
ifIndex                     FROM IF-MIB
sdlcLSAddress               FROM SNA-SDLC-MIB;

```

dlsw MODULE-IDENTITY

```

LAST-UPDATED "9606040900Z"
ORGANIZATION "AIW DLSw MIB RIGLET and IETF DLSw MIB Working Group"
CONTACT-INFO

```

```

"David D. Chen
IBM Corporation
800 Park, Highway 54
Research Triangle Park, NC 27709-9990
Tel: 1 919 254 6182
E-mail: dchen@vnet.ibm.com"

```

DESCRIPTION

```

"This MIB module contains objects to manage Data Link
Switches."

```

::= { mib-2 46 }

```

dlswMIB          OBJECT IDENTIFIER ::= { dlsw 1 }
dlswDomains     OBJECT IDENTIFIER ::= { dlsw 2 }

```

```

-- *****
-- Textual convention definitions
-- *****

```

NBName ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

```

"Represents a single qualified NetBIOS name, which can include
'don't care' and 'wildcard' characters to represent a number
of real NetBIOS names. If an individual character position in
the qualified name contains a '?', the corresponding character
position in a real NetBIOS name is a 'don't care'. If the
qualified name ends in '*', the remainder of a real NetBIOS
name is a 'don't care'. '*' is only considered a wildcard if it
appears at the end of a name."

```

SYNTAX OCTET STRING (SIZE (0..16))

MacAddressNC ::= TEXTUAL-CONVENTION

DISPLAY-HINT "1x:"

STATUS current

DESCRIPTION

"Represents an 802 MAC address represented in

non-canonical format. That is, the most significant bit will be transmitted first. If this information is not available, the value is a zero length string."

SYNTAX OCTET STRING (SIZE (0 | 6))

TAddress ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"Denotes a transport service address.

For dlswTCPDomain, a TAddress is 4 octets long, containing the IP-address in network-byte order."

SYNTAX OCTET STRING (SIZE (0..255))

EndStationLocation ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"Representing the location of an end station related to the managed DLSw node."

SYNTAX INTEGER {  
     other                  (1),  
     internal              (2), -- local virtual MAC address  
     remote                (3), -- via DLSw partner  
     local                  (4) -- locally attached  
 }

DlcType ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"Representing the type of DLC of an end station, if applicable."

SYNTAX INTEGER {  
     other                  (1), -- not assigned yet  
     na                     (2), -- not applicable  
     llc                   (3), -- 802.2 Logical Link Control  
     sdlc                  (4), -- SDLC  
     qllc                  (5) -- QLLC  
 }

LFSize ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"The largest size of the INFO field (including DLC header, not including any MAC-level or framing octets).

64 valid values as defined by the IEEE 802.1D

Addendum are acceptable."

SYNTAX INTEGER {  
     lfs516(516), lfs635(635), lfs754(754), lfs873(873),  
     lfs993(993), lfs1112(1112), lfs1231(1231),  
 }

```

    lfs1350(1350), lfs1470(1470), lfs1542(1542),
    lfs1615(1615), lfs1688(1688), lfs1761(1761),
    lfs1833(1833), lfs1906(1906), lfs1979(1979),
    lfs2052(2052), lfs2345(2345), lfs2638(2638),
    lfs2932(2932), lfs3225(3225), lfs3518(3518),
    lfs3812(3812), lfs4105(4105), lfs4399(4399),
    lfs4865(4865), lfs5331(5331), lfs5798(5798),
    lfs6264(6264), lfs6730(6730), lfs7197(7197),
    lfs7663(7663), lfs8130(8130), lfs8539(8539),
    lfs8949(8949), lfs9358(9358), lfs9768(9768),
    lfs10178(10178), lfs10587(10587), lfs10997(10997),
    lfs11407(11407), lfs12199(12199), lfs12992(12992),
    lfs13785(13785), lfs14578(14578), lfs15370(15370),
    lfs16163(16163), lfs16956(16956), lfs17749(17749),
    lfs20730(20730), lfs23711(23711), lfs26693(26693),
    lfs29674(29674), lfs32655(32655), lfs38618(38618),
    lfs41600(41600), lfs44591(44591), lfs47583(47583),
    lfs50575(50575), lfs53567(53567), lfs56559(56559),
    lfs59551(59551), lfs65535(65535)
}

```

```

null OBJECT IDENTIFIER ::= { 0 0 }

```

```

-- *****
-- DLsw Transport Domain definitions
-- *****

```

```

-- DLsw over TCP

```

```

dlswTCPDomain OBJECT IDENTIFIER ::= { dlswDomains 1 }

```

```

-- for an IP address of length 4:

```

```

--

```

```

-- octets   contents           encoding
-- 1-4      IP-address         network-byte order
--

```

```

DlswTCPAddress ::= TEXTUAL-CONVENTION

```

```

    DISPLAY-HINT "ld.ld.ld.ld"

```

```

    STATUS      current

```

```

    DESCRIPTION

```

```

        "Represents the IP address of a DLsw which uses
        TCP as a transport protocol."

```

```

    SYNTAX      OCTET STRING (SIZE (4))

```

```

-- *****
-- DLsw MIB Definition
-- *****

```

```
-- The DLSw MIB module contains an object part and a conformance part.
-- Object part is organized in the following groups:
-- (1) dlswNode      -- information about this DLSw
-- (2) dlswTConn    -- about adjacent DLSw partners
-- (3) dlswInterface -- about which interfaces DLSw is active on
-- (4) dlswDirectory -- about any directory of local/remote resources
-- (5) dlswCircuit  -- about established circuits.
-- (6) dlswSdlc     -- about SDLC data link switched devices
```

```
dlswNode      OBJECT IDENTIFIER ::= { dlswMIB 1 }
dlswTConn     OBJECT IDENTIFIER ::= { dlswMIB 2 }
dlswInterface OBJECT IDENTIFIER ::= { dlswMIB 3 }
dlswDirectory OBJECT IDENTIFIER ::= { dlswMIB 4 }
dlswCircuit   OBJECT IDENTIFIER ::= { dlswMIB 5 }
dlswSdlc      OBJECT IDENTIFIER ::= { dlswMIB 6 } -- SDLC
```

```
-- *****
-- THE NODE GROUP
-- *****
```

```
-----
-- DLSw Node Identity
-----
```

```
dlswNodeVersion OBJECT-TYPE
SYNTAX      OCTET STRING (SIZE (2))
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "This value identifies the particular version of the DLSw
    standard supported by this DLSw. The first octet is a
    hexadecimal value representing the DLSw standard Version
    number of this DLSw, and the second is a hexadecimal value
    representing the DLSw standard Release number. This
    information is reported in DLSw Capabilities Exchange."
REFERENCE
    "DLSW: Switch-to-Switch Protocol RFC 1795"
 ::= { dlswNode 1 }
```

```
dlswNodeVendorID OBJECT-TYPE
SYNTAX      OCTET STRING (SIZE (3))
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "The value identifies the manufacturer's IEEE-assigned
    organizationally Unique Identifier (OUI) of this DLSw.
    This information is reported in DLSw Capabilities
    Exchange."
REFERENCE
```

```

    "DLSW: Switch-to-Switch Protocol RFC 1795"
 ::= { dlswNode 2 }

```

```
dlswNodeVersionString OBJECT-TYPE
```

```
SYNTAX      DisplayString
```

```
MAX-ACCESS read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```

    "This string gives product-specific information about
     this DLSw (e.g., product name, code release and fix level).
     This flows in Capabilities Exchange messages."

```

```
REFERENCE
```

```

    "DLSW: Switch-to-Switch Protocol RFC 1795"

```

```
 ::= { dlswNode 3 }
```

```

-----
-- DLSw Code Capability
-----

```

```
dlswNodeStdPacingSupport OBJECT-TYPE
```

```
SYNTAX      INTEGER {
```

```

    none                (1),  -- does not support DLSw
                        -- Standard pacing scheme
    adaptiveRcvWindow  (2),  -- the receive window size
                        -- varies
    fixedRcvWindow     (3)   -- the receive window size
                        -- remains constant

```

```
}
```

```
MAX-ACCESS read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```

    "Circuit pacing, as defined in the DLSw Standard, allows each
     of the two DLSw nodes on a circuit to control the amount
     of data the other is permitted to send to them. This object
     reflects the level of support the DLSw node has for this
     protocol. (1) means the node has no support for the standard
     circuit pacing flows; it may use RFC 1434+ methods only, or
     a proprietary flow control scheme. (2) means the node supports
     the standard scheme and can vary the window sizes it grants as
     a data receiver. (3) means the node supports the standard
     scheme but never varies its receive window size."

```

```
 ::= { dlswNode 4 }
```

```

-----
-- DLSw Node Operational Objects
-----

```

```
dlswNodeStatus OBJECT-TYPE
```

```
SYNTAX      INTEGER {
```

```

    active                (1),

```

```

        inactive      (2)
    }
    MAX-ACCESS read-write
    STATUS      current
    DESCRIPTION
        "The status of the DLsw part of the system.  Changing the
        value from active to inactive causes DLsw to take
        the following actions - (1) it disconnects all circuits
        through all DLsw partners, (2) it disconnects all
        transport connections to all DLsw partners, (3) it
        disconnects all local DLC connections, and (4) it stops
        processing all DLC connection set-up traffic.
        Since these are destructive actions, the user should
        query the circuit and transport connection tables in
        advance to understand the effect this action will have.
        Changing the value from inactive to active causes DLsw
        to come up in its initial state, i.e., transport
        connections established and ready to bring up circuits."
 ::= { dlswnode 5 }

```

```

dlswnodeUpTime OBJECT-TYPE
    SYNTAX      TimeTicks
    UNITS       "hundredths of a second"
    MAX-ACCESS read-only
    STATUS      current
    DESCRIPTION
        "The amount of time (in hundredths of a second) since
        the DLsw portion of the system was last re-initialized.
        That is, if dlswnodeState is in the active state,
        the time the dlswnodeState entered the active state.
        It will remain zero if dlswnodeState is in the
        inactive state."
 ::= { dlswnode 6 }

```

```

dlswnodeVirtualSegmentLFSize OBJECT-TYPE
    SYNTAX      LFSize
    MAX-ACCESS read-write
    STATUS      current
    DESCRIPTION
        "The largest frame size (including DLC header and info field
        but not any MAC-level or framing octets) this DLsw can forward
        on any path through itself.  This object can represent any box-
        level frame size forwarding restriction (e.g., from the use
        of fixed-size buffers).  Some DLsw implementations will have
        no such restriction.

        This value will affect the LF size of circuits during circuit
        creation.  The LF size of an existing circuit can be found in

```

```

    the RIF (Routing Information Field)."
```

DEFVAL { lfs65535 }

::= { dlswNode 7 }

```

-- .....
-- NETBIOS Resources
-- .....
```

```

dlswNodeResourceNBExclusivity OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value of true indicates that the NetBIOS Names
        configured in dlswDirNBTable are the only ones accessible
        via this DLSw.

        If a node supports sending run-time capabilities exchange
        messages, changes to this object should cause that action.
        It is up to the implementation exactly when to start the
        run-time capabilities exchange."
    ::= { dlswNode 8 }
```

```

-- .....
-- MAC Address List
-- .....
```

```

dlswNodeResourceMacExclusivity OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value of true indicates that the MAC addresses
        configured in the dlswDirMacTable are the only ones
        accessible via this DLSw.

        If a node supports sending run-time capabilities exchange
        messages, changes to this object should cause that action.
        It is up to the implementation exactly when to start the
        run-time capabilities exchange."
    ::= { dlswNode 9 }
```

```

-- *****
-- TRANSPORT CONNECTION (aka: PARTNER DLSW)
-- *****
```

```

-- -----
```

```
-- Transport Connection Statistics Objects
```

```
-----
dlswTConnStat      OBJECT IDENTIFIER ::= { dlswTConn 1 }
```

```
dlswTConnStatActiveConnections  OBJECT-TYPE
```

```
SYNTAX      Gauge32
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"The number of transport connections that are not in
'disconnected' state."
```

```
::= { dlswTConnStat 1 }
```

```
dlswTConnStatCloseIdles  OBJECT-TYPE
```

```
SYNTAX      Counter32
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"The number of times transport connections in this node
exited the connected state with zero active circuits on
the transport connection."
```

```
::= { dlswTConnStat 2 }
```

```
dlswTConnStatCloseBusys  OBJECT-TYPE
```

```
SYNTAX      Counter32
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"The number of times transport connections in this node
exited the connected state with some non-zero number
of active circuits on the transport connection.  Normally
this means the transport connection failed unexpectedly."
```

```
::= { dlswTConnStat 3 }
```

```
-----
-- Transport Connection Configuration Table
-----
```

```
dlswTConnConfigTable  OBJECT-TYPE
```

```
SYNTAX      SEQUENCE OF DlswTConnConfigEntry
```

```
MAX-ACCESS  not-accessible
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"This table defines the transport connections
that will be initiated or accepted by this
DLSw.  Structure of masks allows wildcard
definition for a collection of transport
connections by a conceptual row.  For a
specific transport connection, there may
```

be multiple of conceptual rows match the transport address. The 'best' match will be the one to determine the characteristics of the transport connection."

```
::= { dlswTConn 2 }
```

```
dlswTConnConfigEntry OBJECT-TYPE
SYNTAX      DlswTConnConfigEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "Each conceptual row defines a collection of
    transport connections."
INDEX       { dlswTConnConfigIndex }
 ::= { dlswTConnConfigTable 1 }
```

```
DlswTConnConfigEntry ::= SEQUENCE {
    dlswTConnConfigIndex          INTEGER,
    dlswTConnConfigTDomain        OBJECT IDENTIFIER,
    dlswTConnConfigLocalTAddr     TAddress,
    dlswTConnConfigRemoteTAddr    TAddress,
    dlswTConnConfigLastModifyTime TimeTicks,
    dlswTConnConfigEntryType      INTEGER,
    dlswTConnConfigGroupDefinition RowPointer,
    dlswTConnConfigSetupType      INTEGER,
    dlswTConnConfigSapList        OCTET STRING,
    dlswTConnConfigAdvertiseMacNB  TruthValue,
    dlswTConnConfigInitCirRecvWndw INTEGER,
    dlswTConnConfigOpens          Counter32,
    dlswTConnConfigRowStatus      RowStatus
}
```

```
dlswTConnConfigIndex OBJECT-TYPE
SYNTAX      INTEGER (0..2147483647)
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The index to the conceptual row of the table.
    Negative numbers are not allowed. There
    are objects defined that point to conceptual
    rows of this table with this index value.
    Zero is used to denote that no corresponding
    row exists.

    Index values are assigned by the agent, and
    should not be reused but should continue to
    increase in value."
 ::= { dlswTConnConfigEntry 1 }
```

```

dlswTConnConfigTDomain OBJECT-TYPE
    SYNTAX      OBJECT IDENTIFIER
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The object identifier which indicates the transport
         domain of this conceptual row."
    ::= { dlswTConnConfigEntry 2 }

dlswTConnConfigLocalTAddr OBJECT-TYPE
    SYNTAX      TAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The local transport address for this conceptual row
         of the transport connection definition."
    ::= { dlswTConnConfigEntry 3 }

dlswTConnConfigRemoteTAddr OBJECT-TYPE
    SYNTAX      TAddress
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The remote transport address. Together with
         dlswTConnConfigEntryType and dlswTConnConfigGroupDefinition,
         the object instance of this conceptual row identifies a
         collection of the transport connections that will be
         either initiated by this DLSw or initiated by a partner
         DLSw and accepted by this DLSw."
    ::= { dlswTConnConfigEntry 4 }

dlswTConnConfigLastModifyTime OBJECT-TYPE
    SYNTAX      TimeTicks
    UNITS       "hundredths of a second"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The time (in hundredths of a second) since the value of
         any object in this conceptual row except for
         dlswTConnConfigOpens was last changed. This value
         may be compared to dlswTConnOperConnectTime to
         determine whether values in this row are completely
         valid for a transport connection created using
         this row definition."
    ::= { dlswTConnConfigEntry 5 }

dlswTConnConfigEntryType OBJECT-TYPE
    SYNTAX      INTEGER {

```

```

        individual      (1),
        global          (2),
        group           (3)
    }

```

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The object instance signifies the type of entry in the associated conceptual row. The value of 'individual' means that the entry applies to a specific partner DLSw node as identified by dlswTConnConfigRemoteTAddr and dlswTConnConfigTDomain. The value of 'global' means that the entry applies to all partner DLSw nodes of the TDomain. The value of 'group' means that the entry applies to a specific set of DLSw nodes in the TDomain. Any group definitions are enterprise-specific and are pointed to by dlswTConnConfigGroupDefinition. In the cases of 'global' and 'group', the value in dlswTConnConfigRemoteTAddr may not have any significance."

```
 ::= { dlswTConnConfigEntry 6 }
```

dlswTConnConfigGroupDefinition OBJECT-TYPE

SYNTAX RowPointer

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"For conceptual rows of 'individual' and 'global' as specified in dlswTConnConfigEntryType, the instance of this object is '0.0'. For conceptual rows of 'group', the instance points to the specific group definition."

```
 ::= { dlswTConnConfigEntry 7 }
```

dlswTConnConfigSetupType OBJECT-TYPE

```

SYNTAX      INTEGER {
    other                (1),
    activePersistent    (2),
    activeOnDemand      (3),
    passive              (4),
    excluded             (5)
}

```

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This value of the instance of a conceptual row identifies the behavior of the collection of transport connections that this conceptual row

defines. The value of activePersistent, activeOnDemand and passive means this DLSw will accept any transport connections, initiated by partner DLSw nodes, which are defined by this conceptual row. The value of activePersistent means this DLSw will also initiate the transport connections of this conceptual row and retry periodically if necessary. The value of activeOnDemand means this DLSw will initiate a transport connection of this conceptual row, if there is a directory cache hits. The value of other is implementation specific. The value of exclude means that the specified node is not allowed to be a partner to this DLSw node. To take a certain conceptual row definition out of service, a value of notInService for dlswTConnConfigRowStatus should be used."

```
DEFVAL { passive }
 ::= { dlswTConnConfigEntry 8 }
```

```
dlswTConnConfigSapList OBJECT-TYPE
SYNTAX      OCTET STRING (SIZE(16))
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
```

"The SAP list indicates which SAPs are advertised to the transport connection defined by this conceptual row. Only SAPs with even numbers are represented, in the form of the most significant bit of the first octet representing the SAP 0, the next most significant bit representing the SAP 2, to the least significant bit of the last octet representing the SAP 254. Data link switching is allowed for those SAPs which have one in its corresponding bit, not allowed otherwise. The whole SAP list has to be changed together. Changing the SAP list affects only new circuit establishments and has no effect on established circuits.

This list can be used to restrict specific partners from knowing about all the SAPs used by DLSw on all its interfaces (these are represented in dlswIfSapList for each interface). For instance, one may want to run NetBIOS with some partners but not others.

If a node supports sending run-time capabilities exchange messages, changes to this object should cause that action. When to start the run-time capabilities exchange is implementation-specific.

The DEFVAL below indicates support for SAPs 0, 4, 8, and C."  
 DEFVAL { 'AA0000000000000000000000000000'H }  
 ::= { dlsWTConnConfigEntry 9 }

dlsWTConnConfigAdvertiseMacNB OBJECT-TYPE

SYNTAX TruthValue  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION

"The value of true indicates that any defined local MAC addresses and NetBIOS names will be advertised to a partner node via initial and (if supported) run-time capabilities exchange messages. The DLSw node should send the appropriate exclusivity control vector to accompany each list it sends, or to represent that the node is explicitly configured to have a null list.

The value of false indicates that the DLSw node should not send a MAC address list or NetBIOS name list, and should also not send their corresponding exclusivity control vectors."

DEFVAL { true }  
 ::= { dlsWTConnConfigEntry 10 }

dlsWTConnConfigInitCirRecvWndw OBJECT-TYPE

SYNTAX INTEGER (0..65535)  
 UNITS "SSP messages"  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION

"The initial circuit receive pacing window size, in the unit of SSP messages, to be used for future transport connections activated using this table row. The managed node sends this value as its initial receive pacing window in its initial capabilities exchange message. Changing this value does not affect the initial circuit receive pacing window size of currently active transport connections. If the standard window pacing scheme is not supported, the value is zero.

A larger receive window value may be appropriate for partners that are reachable only via physical paths that have longer network delays."

DEFVAL { 1 }  
 ::= { dlsWTConnConfigEntry 11 }

dlsWTConnConfigOpens OBJECT-TYPE

SYNTAX Counter32  
 MAX-ACCESS read-only

```

STATUS      current
DESCRIPTION
  "Number of times transport connections entered
  connected state according to the definition of
  this conceptual row."
 ::= { dlswTConnConfigEntry 12 }

```

```

dlswTConnConfigRowStatus  OBJECT-TYPE
SYNTAX      RowStatus
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
  "This object is used by the manager to create
  or delete the row entry in the dlswTConnConfigTable
  following the RowStatus textual convention. The value
  of notInService will be used to take a conceptual
  row definition out of use."
 ::= { dlswTConnConfigEntry 13 }

```

```

-----
-- Transport Connection Operation Table
-----

```

```

-- (1) At most one transport connection can be connected between
--      this DLSw and one of its DLSw partners at a given time.
-- (2) Multiple transport types are supported.
-- (3) Since the entries may be reused, dlswTConnOperEntryTime
--      needs to be consulted for the possibility of counter reset.
-----

```

```

dlswTConnOperTable  OBJECT-TYPE
SYNTAX      SEQUENCE OF DlswTConnOperEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
  "A list of transport connections. It is optional but
  desirable for the agent to keep an entry for some
  period of time after the transport connection is
  disconnected. This allows the manager to capture
  additional useful information about the connection, in
  particular, statistical information and the cause of the
  disconnection."
 ::= { dlswTConn 3 }

```

```

dlswTConnOperEntry  OBJECT-TYPE
SYNTAX      DlswTConnOperEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION

```

""

```
INDEX { dlswTConnOperTDomain, dlswTConnOperRemoteTAddr }
 ::= { dlswTConnOperTable 1 }
```

```
DlswTConnOperEntry ::= SEQUENCE {
  dlswTConnOperTDomain          OBJECT IDENTIFIER,
  dlswTConnOperLocalTAddr      TAddress,
  dlswTConnOperRemoteTAddr     TAddress,

  dlswTConnOperEntryTime       TimeTicks,
  dlswTConnOperConnectTime     TimeTicks,
  dlswTConnOperState           INTEGER,
  dlswTConnOperConfigIndex     INTEGER,
  dlswTConnOperFlowCntlMode    INTEGER,

  dlswTConnOperPartnerVersion  OCTET STRING,
  dlswTConnOperPartnerVendorID OCTET STRING,
  dlswTConnOperPartnerVersionStr DisplayString,
  dlswTConnOperPartnerInitPacingWndw INTEGER,

  dlswTConnOperPartnerSapList  OCTET STRING,
  dlswTConnOperPartnerNBExcl   TruthValue,
  dlswTConnOperPartnerMacExcl  TruthValue,
  dlswTConnOperPartnerNBInfo   INTEGER,
  dlswTConnOperPartnerMacInfo  INTEGER,

  dlswTConnOperDiscTime        TimeTicks,
  dlswTConnOperDiscReason      INTEGER,
  dlswTConnOperDiscActiveCir   INTEGER,

  dlswTConnOperInDataPkts      Counter32,
  dlswTConnOperOutDataPkts     Counter32,
  dlswTConnOperInDataOctets    Counter32,
  dlswTConnOperOutDataOctets   Counter32,

  dlswTConnOperInCntlPkts      Counter32,
  dlswTConnOperOutCntlPkts     Counter32,

  dlswTConnOperCURExSents      Counter32,
  dlswTConnOperICRexRcvds     Counter32,
  dlswTConnOperCURExRcvds     Counter32,
  dlswTConnOperICRexSents     Counter32,

  dlswTConnOperNQexSents      Counter32,
  dlswTConnOperNRexRcvds     Counter32,
  dlswTConnOperNQexRcvds     Counter32,
  dlswTConnOperNRexSents     Counter32,
```

```

dlswTConnOperCirCreates      Counter32,
dlswTConnOperCircuits      Gauge32
}

```

```

dlswTConnOperTDomain OBJECT-TYPE
SYNTAX      OBJECT IDENTIFIER
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The object identifier indicates the transport domain
    of this transport connection."
 ::= { dlswTConnOperEntry 1 }

```

```

dlswTConnOperLocalTAddr OBJECT-TYPE
SYNTAX      TAddress
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "The local transport address for this transport connection.
    This value could be different from dlswTConnConfigLocalAddr,
    if the value of the latter were changed after this transport
    connection was established."
 ::= { dlswTConnOperEntry 2 }

```

```

dlswTConnOperRemoteTAddr OBJECT-TYPE
SYNTAX      TAddress
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The remote transport address of this transport connection."
 ::= { dlswTConnOperEntry 3 }

```

```

dlswTConnOperEntryTime OBJECT-TYPE
SYNTAX      TimeTicks
UNITS       "hundredths of a second"
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "The amount of time (in hundredths of a second) since this
    transport connection conceptual row was created."
 ::= { dlswTConnOperEntry 4 }

```

```

-- .....
-- DLSw Transport Connection Operational Objects
-- .....

```

```

dlswTConnOperConnectTime OBJECT-TYPE
SYNTAX      TimeTicks

```

```

UNITS          "hundredths of a second"
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
  "The amount of time (in hundredths of a second) since this
  transport connection last entered the 'connected' state.
  A value of zero means this transport connection has never
  been established."
 ::= { dlswTConnOperEntry 5 }

```

```
dlswTConnOperState OBJECT-TYPE
```

```

SYNTAX        INTEGER {
  connecting          (1),
  initCapExchange    (2),
  connected           (3),
  quiescing           (4),
  disconnecting      (5),
  disconnected         (6)
}

```

```
MAX-ACCESS    read-write
```

```
STATUS        current
```

```
DESCRIPTION
```

"The state of this transport connection. The transport connection enters 'connecting' state when DLSw makes a connection request to the transport layer. Once initial Capabilities Exchange is sent, the transport connection enters 'initCapExchange' state. When partner capabilities have been determined and the transport connection is ready for sending CanUReach (CUR) messages, it moves to the 'connected' state. When DLSw is in the process of bringing down the connection, it is in the 'disconnecting' state. When the transport layer indicates one of its connections is disconnected, the transport connection moves to the 'disconnected' state.

Whereas all of the values will be returned in response to a management protocol retrieval operation, only two values may be specified in a management protocol set operation: 'quiescing' and 'disconnecting'. Changing the value to 'quiescing' prevents new circuits from being established, and will cause a transport disconnect when the last circuit on the connection goes away. Changing the value to 'disconnecting' will force off all circuits immediately and bring the connection to 'disconnected' state."

```
 ::= { dlswTConnOperEntry 6 }

```

```
dlswTConnOperConfigIndex OBJECT-TYPE
```

```

SYNTAX      INTEGER (0..2147483647)
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION

```

"The value of dlswTConnConfigIndex of the dlswTConnConfigEntry that governs the configuration information used by this dlswTConnOperEntry. The manager can therefore normally examine both configured and operational information for this transport connection.

This value is zero if the corresponding dlswTConnConfigEntry was deleted after the creation of this dlswTConnOperEntry. If some fields in the former were changed but the conceptual row was not deleted, some configuration information may not be valid for this operational transport connection. The manager can compare dlswTConnOperConnectTime and dlswTConnConfigLastModifyTime to determine if this condition exists."

```
 ::= { dlswTConnOperEntry 7 }
```

```

-- .....
-- Transport Connection Characteristics
-- .....

```

```
dlswTConnOperFlowCntlMode OBJECT-TYPE
```

```

SYNTAX      INTEGER {
    undetermined (1),
    pacing       (2),  -- DLSw standard flow control
    other        (3)  -- non-DLSw standard flow control
}

```

```

MAX-ACCESS  read-only
STATUS      current

```

```
DESCRIPTION
```

"The flow control mechanism in use on this transport connection. This value is undetermined (1) before the mode of flow control can be established on a new transport connection (i.e., after CapEx is sent but before Capex or other SSP control messages have been received). Pacing (2) indicates that the standard RFC 1795 pacing mechanism is in use. Other (3) may be either the RFC 1434+ xBusy mechanism operating to a back-level DLSw, or a vendor-specific flow control method. Whether it is xBusy or not can be inferred from dlswTConnOperPartnerVersion."

```
 ::= { dlswTConnOperEntry 8 }
```

```

-- .....

```

```
dlswTConnOperPartnerVersion OBJECT-TYPE
```

```

SYNTAX      OCTET STRING (SIZE (0 | 2))

```

MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"This value identifies which version (first octet) and release (second octet) of the DLSw standard is supported by this partner DLSw. This information is obtained from a DLSw capabilities exchange message received from the partner DLSw. A string of zero length is returned before a Capabilities Exchange message is received, or if one is never received. A conceptual row with a dlswTConnOperState of 'connected' but a zero length partner version indicates that the partner is a non-standard DLSw partner.

If an implementation chooses to keep dlswTConnOperEntry in the 'disconnected' state, this value should remain unchanged."

REFERENCE

"DLSW: Switch-to-Switch Protocol RFC 1795"

::= { dlswTConnOperEntry 9 }

dlswTConnOperPartnerVendorID OBJECT-TYPE  
 SYNTAX OCTET STRING (SIZE (0 | 3))  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"This value identifies the IEEE-assigned organizationally Unique Identifier (OUI) of the maker of this partner DLSw. This information is obtained from a DLSw capabilities exchange message received from the partner DLSw. A string of zero length is returned before a Capabilities Exchange message is received, or if one is never received.

If an implementation chooses to keep dlswTConnOperEntry in the 'disconnected' state, this value should remain unchanged."

::= { dlswTConnOperEntry 10 }

dlswTConnOperPartnerVersionStr OBJECT-TYPE  
 SYNTAX DisplayString (SIZE (0..253))  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"This value identifies the particular product version (e.g., product name, code level, fix level) of this partner DLSw. The format of the actual version string is vendor-specific. This information is obtained from a DLSw capabilities exchange message received from the partner DLSw. A string of zero length is returned before a Capabilities Exchange message is received, if one is never received, or if one is received but it does not contain a version string.

If an implementation chooses to keep dlswTConnOperEntry in the 'disconnected' state, this value should remain unchanged."

REFERENCE

"DLSW: Switch-to-Switch Protocol RFC 1795"

::= { dlswTConnOperEntry 11 }

dlswTConnOperPartnerInitPacingWndw OBJECT-TYPE

SYNTAX INTEGER (0..65535)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of the partner initial receive pacing window. This is our initial send pacing window for all new circuits on this transport connection, as modified and granted by the first flow control indication the partner sends on each circuit. This information is obtained from a DLSw capabilities exchange message received from the partner DLSw.

A value of zero is returned before a Capabilities Exchange message is received, or if one is never received.

If an implementation chooses to keep dlswTConnOperEntry in the 'disconnected' state, this value should remain unchanged."

REFERENCE

"DLSW: Switch-to-Switch Protocol RFC 1795"

::= { dlswTConnOperEntry 12 }

-- .....

dlswTConnOperPartnerSapList OBJECT-TYPE

SYNTAX OCTET STRING (SIZE (0 | 16))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The Supported SAP List received in the capabilities exchange message from the partner DLSw. This list has the same format described for dlswTConnConfigSapList. A string of zero length is returned before a Capabilities Exchange message is received, or if one is never received.

If an implementation chooses to keep dlswTConnOperEntry in the 'disconnected' state, this value should remain unchanged."

::= { dlswTConnOperEntry 13 }

dlswTConnOperPartnerNBExcl OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of true signifies that the NetBIOS names received from this partner in the NetBIOS name list in its capabilities exchange message are the only NetBIOS names reachable by that partner. 'False' indicates that other NetBIOS names may be reachable. 'False' should be returned before a Capabilities Exchange message is received, if one is never received, or if one is received without a NB Name Exclusivity CV.

If an implementation chooses to keep dlswTConnOperEntry in the 'disconnected' state, this value should remain unchanged."  
 ::= { dlswTConnOperEntry 14 }

dlswTConnOperPartnerMacExcl OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of true signifies that the MAC addresses received from this partner in the MAC address list in its capabilities exchange message are the only MAC addresses reachable by that partner. 'False' indicates that other MAC addresses may be reachable. 'False' should be returned before a Capabilities Exchange message is received, if one is never received, or if one is received without a MAC Address Exclusivity CV.

If an implementation chooses to keep dlswTConnOperEntry in the 'disconnected' state, this value should remain unchanged."  
 ::= { dlswTConnOperEntry 15 }

dlswTConnOperPartnerNBInfo OBJECT-TYPE

SYNTAX INTEGER {  
   none (1), -- none is kept  
   partial (2), -- partial list is kept  
   complete (3), -- complete list is kept  
   notApplicable (4)  
 }

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"It is up to this DSLw whether to keep either none, some, or all of the NetBIOS name list that was received in the capabilities exchange message sent by this partner DLSw. This object identifies how much information was kept by this DLSw. These names are stored as userConfigured remote entries in dlswDirNBTable.  
 A value of (4), notApplicable, should be returned before a Capabilities Exchange message is received, or if one is never received.

If an implementation chooses to keep dlswTConnOperEntry in the 'disconnected' state, this value should remain unchanged."  
 ::= { dlswTConnOperEntry 16 }

dlswTConnOperPartnerMacInfo OBJECT-TYPE

SYNTAX INTEGER {  
 none (1), -- none is kept  
 partial (2), -- partial list is kept  
 complete (3), -- complete list is kept  
 notApplicable (4)  
 }

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"It is up to this DLSw whether to keep either none, some, or all of the MAC address list that was received in the capabilities exchange message sent by this partner DLSw. This object identifies how much information was kept by this DLSw. These names are stored as userConfigured remote entries in dlswDirMACTable.  
 A value of (4), notApplicable, should be returned before a Capabilities Exchange message is received, or if one is never received.

If an implementation chooses to keep dlswTConnOperEntry in the 'disconnected' state, this value should remain unchanged."  
 ::= { dlswTConnOperEntry 17 }

-- .....  
 -- Information about the last disconnect of this transport connection.  
 -- These objects make sense only for implementations that keep  
 -- transport connection information around after disconnection.  
 -- .....

dlswTConnOperDiscTime OBJECT-TYPE

SYNTAX TimeTicks  
 UNITS "hundredths of a second"  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION

"The amount of time (in hundredths of a second) since the dlswTConnOperState last entered 'disconnected' state."

::= { dlswTConnOperEntry 18 }

dlswTConnOperDiscReason OBJECT-TYPE

SYNTAX INTEGER {  
 other (1),  
 capExFailed (2),  
 transportLayerDisc (3),  
 }

```

        operatorCommand      (4),
        lastCircuitDiscd    (5),
        protocolError       (6)
    }

```

```

MAX-ACCESS read-only
STATUS      current
DESCRIPTION

```

```

    "This object signifies the reason that either prevented the
    transport connection from entering the connected state, or
    caused the transport connection to enter the disconnected
    state."

```

```

 ::= { dlswTConnOperEntry 19 }

```

```

dlswTConnOperDiscActiveCir OBJECT-TYPE

```

```

SYNTAX      INTEGER (0..2147483647)
MAX-ACCESS read-only
STATUS      current
DESCRIPTION

```

```

    "The number of circuits active (not in DISCONNECTED state)
    at the time the transport connection was last disconnected.
    This value is zero if the transport connection has never
    been connected."

```

```

 ::= { dlswTConnOperEntry 20 }

```

```

-- .....
-- Transport Connection Statistics
-- (1) Traffic counts
-- .....

```

```

dlswTConnOperInDataPkts OBJECT-TYPE

```

```

SYNTAX      Counter32
UNITS       "SSP messages"
MAX-ACCESS read-only
STATUS      current
DESCRIPTION

```

```

    "The number of Switch-to-Switch Protocol (SSP) messages of
    type DGRMFRAME, DATAFRAME, or INFOFRAME received on this
    transport connection."

```

```

 ::= { dlswTConnOperEntry 21 }

```

```

dlswTConnOperOutDataPkts OBJECT-TYPE

```

```

SYNTAX      Counter32
UNITS       "SSP messages"
MAX-ACCESS read-only
STATUS      current
DESCRIPTION

```

```

    "The number of Switch-to-Switch Protocol (SSP) messages of
    type DGRMFRAME, DATAFRAME, or INFOFRAME transmitted on this
    transport connection."

```

```
::= { dlswTConnOperEntry 22 }
```

```
dlswTConnOperInDataOctets OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
UNITS "octets"
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"The number octets in Switch-to-Switch Protocol (SSP) messages of type DGRMFRAME, DATAFRAME, or INFOFRAME received on this transport connection. Each message is counted starting with the first octet following the SSP message header."
```

```
::= { dlswTConnOperEntry 23 }
```

```
dlswTConnOperOutDataOctets OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
UNITS "octets"
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"The number octets in Switch-to-Switch Protocol (SSP) messages of type DGRMFRAME, DATAFRAME, or INFOFRAME transmitted on this transport connection. Each message is counted starting with the first octet following the SSP message header."
```

```
::= { dlswTConnOperEntry 24 }
```

```
dlswTConnOperInCntlPkts OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
UNITS "SSP messages"
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"The number of Switch-to-Switch Protocol (SSP) messages received on this transport connection which were not of type DGRMFRAME, DATAFRAME, or INFOFRAME."
```

```
::= { dlswTConnOperEntry 25 }
```

```
dlswTConnOperOutCntlPkts OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
UNITS "SSP messages"
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"The number of Switch-to-Switch Protocol (SSP) messages of transmitted on this transport connection which were not of type DGRMFRAME, DATAFRAME, or INFOFRAME."
```

```
::= { dlswTConnOperEntry 26 }
```

```
-- .....
-- (2) Directory activities (Explorer messages)
-- .....
```

```
dlswTConnOperCURExSents OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of CanUReach_ex messages sent on this transport
        connection."
    ::= { dlswTConnOperEntry 27 }
```

```
dlswTConnOperICRexRcvds OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of ICanReach_ex messages received on this transport
        connection."
    ::= { dlswTConnOperEntry 28 }
```

```
dlswTConnOperCURExRcvds OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of CanUReach_ex messages received on this transport
        connection."
    ::= { dlswTConnOperEntry 29 }
```

```
dlswTConnOperICRexSents OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of ICanReach_ex messages sent on this transport
        connection."
    ::= { dlswTConnOperEntry 30 }
```

```
-- .....
```

```
dlswTConnOperNQexSents OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of NetBIOS_NQ_ex (NetBIOS Name Query-explorer)
```

messages sent on this transport connection."  
 ::= { dlswTConnOperEntry 31 }

dlswTConnOperNRexRcvds OBJECT-TYPE  
 SYNTAX Counter32  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION  
 "The number of NETBIOS\_NR\_ex (NetBIOS Name Recognized-explorer)  
 messages received on this transport connection."  
 ::= { dlswTConnOperEntry 32 }

dlswTConnOperNQexRcvds OBJECT-TYPE  
 SYNTAX Counter32  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION  
 "The number of NETBIOS\_NQ\_ex messages received on this  
 transport connection."  
 ::= { dlswTConnOperEntry 33 }

dlswTConnOperNRexSents OBJECT-TYPE  
 SYNTAX Counter32  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION  
 "The number of NETBIOS\_NR\_ex messages sent on this transport  
 connection."  
 ::= { dlswTConnOperEntry 34 }

-- .....  
 -- (3) Circuit activities on each transport connection  
 -- .....

dlswTConnOperCirCreates OBJECT-TYPE  
 SYNTAX Counter32  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION  
 "The number of times that circuits entered 'circuit\_established'  
 state (not counting transitions from 'circuit\_restart')."  
 ::= { dlswTConnOperEntry 35 }

dlswTConnOperCircuits OBJECT-TYPE  
 SYNTAX Gauge32  
 MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION  
 "The number of currently active circuits on this transport

connection, where 'active' means not in 'disconnected' state."  
 ::= { dlswTConnOperEntry 36 }

-----  
 -- Transport Connection Specific  
 -----

dlswTConnSpecific OBJECT IDENTIFIER ::= { dlswTConn 4 }  
 dlswTConnTcp OBJECT IDENTIFIER ::= { dlswTConnSpecific 1 }

-----  
 -- TCP Transport Connection Specific -- Configuration  
 -----

dlswTConnTcpConfigTable OBJECT-TYPE  
 SYNTAX SEQUENCE OF DlswTConnTcpConfigEntry  
 MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION  
 "This table defines the TCP transport connections that  
 will be either initiated by or accepted by this DSLw.  
 It augments the entries in dlswTConnConfigTable whose domain  
 is dlswTCPDomain."  
 ::= { dlswTConnTcp 1 }

dlswTConnTcpConfigEntry OBJECT-TYPE  
 SYNTAX DlswTConnTcpConfigEntry  
 MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION  
 "Each conceptual row defines parameters that are  
 specific to dlswTCPDomain transport connections."  
 INDEX { dlswTConnConfigIndex }  
 ::= { dlswTConnTcpConfigTable 1 }

DlswTConnTcpConfigEntry ::= SEQUENCE {  
 dlswTConnTcpConfigKeepAliveInt INTEGER,  
 dlswTConnTcpConfigTcpConnections INTEGER,  
 dlswTConnTcpConfigMaxSegmentSize INTEGER  
 }

dlswTConnTcpConfigKeepAliveInt OBJECT-TYPE  
 SYNTAX INTEGER (0..1800)  
 UNITS "seconds"  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION  
 "The time in seconds between TCP keepAlive messages when  
 no traffic is flowing. Zero signifies no keepAlive protocol."

Changes take effect only for new TCP connections."  
 DEFVAL { 0 }  
 ::= { dlswTConnTcpConfigEntry 1 }

dlswTConnTcpConfigTcpConnections OBJECT-TYPE  
 SYNTAX INTEGER (1..16)  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION  
 "This is our preferred number of TCP connections within a  
 TCP transport connection. The actual number used is negotiated  
 at capabilities exchange time. Changes take effect only  
 for new transport connections."  
 DEFVAL { 2 }  
 ::= { dlswTConnTcpConfigEntry 2 }

dlswTConnTcpConfigMaxSegmentSize OBJECT-TYPE  
 SYNTAX INTEGER (0..65535)  
 UNITS "packets"  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION  
 "This is the number of bytes that this node is  
 willing to receive over the read TCP connection(s).  
 Changes take effect for new transport connections."  
 DEFVAL { 4096 }  
 ::= { dlswTConnTcpConfigEntry 3 }

-- .....  
 -- TCP Transport Connection Specific -- Operation  
 -- .....

dlswTConnTcpOperTable OBJECT-TYPE  
 SYNTAX SEQUENCE OF DlswTConnTcpOperEntry  
 MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION  
 "A list of TCP transport connections. It is optional  
 but desirable for the agent to keep an entry for some  
 period of time after the transport connection is  
 disconnected. This allows the manager to capture  
 additional useful information about the connection, in  
 particular, statistical information and the cause of the  
 disconnection."  
 ::= { dlswTConnTcp 2 }

dlswTConnTcpOperEntry OBJECT-TYPE  
 SYNTAX DlswTConnTcpOperEntry

MAX-ACCESS not-accessible  
STATUS current  
DESCRIPTION

""

INDEX { dlswTConnOperTDomain, dlswTConnOperRemoteTAddr }  
 ::= { dlswTConnTcpOperTable 1 }

dlswTConnTcpOperEntry ::= SEQUENCE {  
 dlswTConnTcpOperKeepAliveInt INTEGER,  
 dlswTConnTcpOperPrefTcpConnections INTEGER,  
 dlswTConnTcpOperTcpConnections INTEGER  
 }

dlswTConnTcpOperKeepAliveInt OBJECT-TYPE

SYNTAX INTEGER (0..1800)

UNITS "seconds"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The time in seconds between TCP keepAlive messages when  
 no traffic is flowing. Zero signifies no keepAlive protocol is  
 operating."

::= { dlswTConnTcpOperEntry 1 }

dlswTConnTcpOperPrefTcpConnections OBJECT-TYPE

SYNTAX INTEGER (1..16)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This is the number of TCP connections preferred by this DLSw  
 partner, as received in its capabilities exchange message."

::= { dlswTConnTcpOperEntry 2 }

dlswTConnTcpOperTcpConnections OBJECT-TYPE

SYNTAX INTEGER (1..16)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This is the actual current number of TCP connections within  
 this transport connection."

::= { dlswTConnTcpOperEntry 3 }

-- \*\*\*\*\*  
-- DLSW INTERFACE GROUP  
-- \*\*\*\*\*

dlswIfTable OBJECT-TYPE

```

SYNTAX      SEQUENCE OF DlswIfEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The list of interfaces on which DLSw is active."
 ::= { dlswInterface 1 }

```

```

dlswIfEntry OBJECT-TYPE
SYNTAX      DlswIfEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    ""
INDEX       { ifIndex }
 ::= { dlswIfTable 1 }

```

```

DlswIfEntry ::= SEQUENCE {
    dlswIfRowStatus      RowStatus,
    dlswIfVirtualSegment INTEGER,
    dlswIfSapList        OCTET STRING
}

```

```

dlswIfRowStatus OBJECT-TYPE
SYNTAX          RowStatus
MAX-ACCESS      read-create
STATUS          current
DESCRIPTION
    "This object is used by the manager to create
    or delete the row entry in the dlswIfTable
    following the RowStatus textual convention."
 ::= { dlswIfEntry 1 }

```

```

dlswIfVirtualSegment OBJECT-TYPE
SYNTAX          INTEGER (0..4095 | 65535)
MAX-ACCESS      read-create
STATUS          current
DESCRIPTION
    "The segment number that uniquely identifies the virtual
    segment to which this DLSw interface is connected.
    Current source routing protocols limit this value to
    the range 0 - 4095. (The value 0 is used by some
    management applications for special test cases.)
    A value of 65535 signifies that no virtual segment
    is assigned to this interface. For instance,
    in a non-source routing environment, segment number
    assignment is not required."
DEFVAL { 65535 }
 ::= { dlswIfEntry 2 }

```

```

dlswIfSapList OBJECT-TYPE
  SYNTAX      OCTET STRING  (SIZE(16))
  MAX-ACCESS  read-create
  STATUS      current
  DESCRIPTION

```

"The SAP list indicates which SAPs are allowed to be data link switched through this interface. This list has the same format described for dlswTConnConfigSapList.

When changes to this object take effect is implementation-specific. Turning off a particular SAP can destroy active circuits that are using that SAP. An agent implementation may reject such changes until there are no active circuits if it so chooses. In this case, it is up to the manager to close the circuits first, using dlswCircuitState.

The DEFVAL below indicates support for SAPs 0, 4, 8, and C."

```

DEFVAL { 'AA00000000000000000000000000000000'H }
 ::= { dlswIfEntry 3 }

```

```

-- *****
-- DIRECTORY
-- Directory services caches the locations of MAC addresses
-- and NetBIOS names. For resources which are attached via
-- local interfaces, the ifIndex may be cached, and for
-- resources which are reachable via a DLSw partner, the
-- transport address of the DLSw partner is cached.
-- *****

```

```

-- -----
-- Directory Related Statistical Objects
-- -----

```

```

dlswDirStat      OBJECT IDENTIFIER ::= { dlswDirectory 1 }

```

```

dlswDirMacEntries OBJECT-TYPE
  SYNTAX      Gauge32
  MAX-ACCESS  read-only
  STATUS      current
  DESCRIPTION

```

"The current total number of entries in the dlswDirMacTable."  
 ::= { dlswDirStat 1 }

```

dlswDirMacCacheHits OBJECT-TYPE
  SYNTAX      Counter32
  MAX-ACCESS  read-only
  STATUS      current

```

## DESCRIPTION

"The number of times a cache search for a particular MAC address resulted in success."

::= { dlswDirStat 2 }

## dlswDirMacCacheMisses OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The number of times a cache search for a particular MAC address resulted in failure."

::= { dlswDirStat 3 }

## dlswDirMacCacheNextIndex OBJECT-TYPE

SYNTAX INTEGER (0..2147483647)

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The next value of dlswDirMacIndex to be assigned by the agent. A retrieval of this object atomically reserves the returned value for use by the manager to create a row in dlswDirMacTable. This makes it possible for the agent to control the index space of the MAC address cache, yet allows the manager to administratively create new rows."

::= { dlswDirStat 4 }

## dlswDirNBEntries OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The current total number of entries in the dlswDirNBTable."

::= { dlswDirStat 5 }

## dlswDirNBCacheHits OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The number of times a cache search for a particular NetBIOS name resulted in success."

::= { dlswDirStat 6 }

## dlswDirNBCacheMisses OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times a cache search for a particular NetBIOS name resulted in failure."

::= { dlswDirStat 7 }

dlswDirNBCacheNextIndex OBJECT-TYPE

SYNTAX INTEGER (0..2147483647)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The next value of dlswDirNBIndex to be assigned by the agent. A retrieval of this object atomically reserves the returned value for use by the manager to create a row in dlswDirNBTable. This makes it possible for the agent to control the index space for the NetBIOS name cache, yet allows the manager to administratively create new rows."

::= { dlswDirStat 8 }

-- -----  
-- Directory Cache  
-- -----

dlswDirCache OBJECT IDENTIFIER ::= { dlswDirectory 2 }

-- .....  
-- Directory for MAC Addresses.  
-- All Possible combinations of values of these objects.  
--

EntryType	LocationType	Location	Status
userConfigured	local	ifEntry or 0.0	reachable, or notReachable, or unknown
userConfigured	remote	TConnConfigEntry	reachable, or notReachable, or unknown
partnerCapExMsg	remote	TConnOperEntry	unknown
dynamic	local	ifEntry or 0.0	reachable
dynamic	remote	TConnOperEntry	reachable

dlswDirMacTable OBJECT-TYPE

SYNTAX SEQUENCE OF DlswDirMacEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table contains locations of MAC addresses. They could be either verified or not verified,

local or remote, and configured locally or learned from either Capabilities Exchange messages or directory searches."

::= { dlswDirCache 1 }

dlswDirMacEntry OBJECT-TYPE  
 SYNTAX DlswDirMacEntry  
 MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION  
 "Indexed by dlswDirMacIndex."  
 INDEX { dlswDirMacIndex }  
 ::= { dlswDirMacTable 1 }

DlswDirMacEntry ::= SEQUENCE {  
 dlswDirMacIndex INTEGER,  
 dlswDirMacMac MacAddressNC,  
 dlswDirMacMask MacAddressNC,  
 dlswDirMacEntryType INTEGER,  
 dlswDirMacLocationType INTEGER,  
 dlswDirMacLocation RowPointer,  
 dlswDirMacStatus INTEGER,  
 dlswDirMacLFSize LFSize,  
 dlswDirMacRowStatus RowStatus  
 }

dlswDirMacIndex OBJECT-TYPE  
 SYNTAX INTEGER (0..2147483647)  
 MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION  
 "Uniquely identifies a conceptual row of this table."  
 ::= { dlswDirMacEntry 1 }

dlswDirMacMac OBJECT-TYPE  
 SYNTAX MacAddressNC  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION  
 "The MAC address, together with the dlswDirMacMask, specifies a set of MAC addresses that are defined or discovered through an interface or partner DLSw nodes."  
 ::= { dlswDirMacEntry 2 }

dlswDirMacMask OBJECT-TYPE  
 SYNTAX MacAddressNC  
 MAX-ACCESS read-create  
 STATUS current

## DESCRIPTION

"The MAC address mask, together with the dlswDirMacMac, specifies a set of MAC addresses that are defined or discovered through an interface or partner DLSw nodes."

DEFVAL { 'FFFFFFFFFFFF'H }  
 ::= { dlswDirMacEntry 3 }

dlswDirMacEntryType OBJECT-TYPE

SYNTAX INTEGER {  
     other (1),  
     userConfiguredPublic (2),  
     userConfiguredPrivate (3),  
     partnerCapExMsg (4),  
     dynamic (5)  
 }

MAX-ACCESS read-create

STATUS current

## DESCRIPTION

"The cause of the creation of this conceptual row. It could be one of the three methods: (1) user configured, including via management protocol set operations, configuration file, command line or equivalent methods; (2) learned from the partner DLSw Capabilities Exchange messages; and (3) dynamic, e.g., learned from ICanReach messages, or LAN explorer frames. Since only individual MAC addresses can be dynamically learned, dynamic entries will all have a mask of all FFs.

The public versus private distinction for user-configured resources applies only to local resources (UC remote resources are private), and indicates whether that resource should be advertised in capabilities exchange messages sent by this node."

DEFVAL { userConfiguredPublic }  
 ::= { dlswDirMacEntry 4 }

dlswDirMacLocationType OBJECT-TYPE

SYNTAX INTEGER {  
     other (1),  
     local (2),  
     remote (3)  
 }

MAX-ACCESS read-create

STATUS current

## DESCRIPTION

"The location of the resource (or a collection of resources using a mask) of this conceptual row

is either (1) local - the resource is reachable via an interface, or (2) remote - the resource is reachable via a partner DLSw node (or a set of partner DLSw nodes)."

DEFVAL { local }  
 ::= { dlswDirMacEntry 5 }

dlswDirMacLocation OBJECT-TYPE

SYNTAX RowPointer  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION

"Points to either the ifEntry, dlswTConnConfigEntry, dlswTConnOperEntry, 0.0, or something that is implementation specific. It identifies the location of the MAC address (or the collection of MAC addresses.)"

DEFVAL { null }  
 ::= { dlswDirMacEntry 6 }

dlswDirMacStatus OBJECT-TYPE

SYNTAX INTEGER {  
     unknown                    (1),  
     reachable                  (2),  
     notReachable              (3)  
 }

MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION

"This object specifies whether DLSw currently believes the MAC address to be accessible at the specified location. The value 'notReachable' allows a configured resource definition to be taken out of service when a search to that resource fails (avoiding a repeat of the search)."

DEFVAL { unknown }  
 ::= { dlswDirMacEntry 7 }

dlswDirMacLFSsize OBJECT-TYPE

SYNTAX LFSsize  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION

"The largest size of the MAC INFO field (LLC header and data) that a circuit to the MAC address can carry through this path."

DEFVAL { lfs65535 }  
 ::= { dlswDirMacEntry 8 }

dlswDirMacRowStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create
STATUS current
DESCRIPTION

"This object is used by the manager to create
or delete the row entry in the dlswDirMacTable
following the RowStatus textual convention."

::= { dlswDirMacEntry 9 }

-- .....
-- Directory for NetBIOS Names
-- All Possible combinations of values of these objects.

Table with 4 columns: EntryType, LocationType, Location, Status. Rows include userConfigured (local, ifEntry or 0.0), userConfigured (remote, TConnConfigEntry), partnerCapExMsg (remote, TConnOperEntry), dynamic (local, ifEntry or 0.0), and dynamic (remote, TConnOperEntry).

dlswDirNBTable OBJECT-TYPE
SYNTAX SEQUENCE OF DlswDirNBEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION

"This table contains locations of NetBIOS names.
They could be either verified or not verified,
local or remote, and configured locally or learned
from either Capabilities Exchange messages or
directory searches."

::= { dlswDirCache 2 }

dlswDirNBEntry OBJECT-TYPE
SYNTAX DlswDirNBEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION

"Indexed by dlswDirNBIndex."

INDEX { dlswDirNBIndex }
::= { dlswDirNBTable 1 }

DlswDirNBEntry ::= SEQUENCE {
dlswDirNBIndex INTEGER,

```

dlswDirNBName          NBName,
dlswDirNBNameType     INTEGER,
dlswDirNBEntryType    INTEGER,
dlswDirNBLocationType INTEGER,
dlswDirNBLocation     RowPointer,
dlswDirNBStatus       INTEGER,
dlswDirNBLFSize       LFSize,
dlswDirNBRowStatus    RowStatus
}

```

```

dlswDirNBIndex OBJECT-TYPE
    SYNTAX      INTEGER (0..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Uniquely identifies a conceptual row of this table."
    ::= { dlswDirNBEntry 1 }

```

```

dlswDirNBName OBJECT-TYPE
    SYNTAX      NBName
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The NetBIOS name (including 'any char' and 'wildcard'
        characters) specifies a set of NetBIOS names that are
        defined or discovered through an interface or partner
        DLSw nodes."
    ::= { dlswDirNBEntry 2 }

```

```

dlswDirNBNameType OBJECT-TYPE
    SYNTAX      INTEGER {
        unknown      (1),
        individual   (2),
        group        (3)
    }
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "Whether dlswDirNBName represents an (or a set of) individual
        or group NetBIOS name(s)."
    DEFVAL { unknown }
    ::= { dlswDirNBEntry 3 }

```

```

dlswDirNBEntryType OBJECT-TYPE
    SYNTAX      INTEGER {
        other                (1),
        userConfiguredPublic (2),
        userConfiguredPrivate (3),
    }

```

```

    partnerCapExMsg      (4),
    dynamic              (5)
}
MAX-ACCESS read-create
STATUS      current
DESCRIPTION

```

"The cause of the creation of this conceptual row. It could be one of the three methods: (1) user configured, including via management protocol set operations, configuration file, command line, or equivalent methods; (2) learned from the partner DLSw Capabilities Exchange messages; and (3) dynamic, e.g., learned from ICanReach messages, or test frames. Since only actual NetBIOS names can be dynamically learned, dynamic entries will not contain any char or wildcard characters.

The public versus private distinction for user-configured resources applies only to local resources (UC remote resources are private), and indicates whether that resource should be advertised in capabilities exchange messages sent by this node."

```

DEFVAL { userConfiguredPublic }
 ::= { dlswDirNBEntry 4 }

```

```

dlswDirNBLocationType OBJECT-TYPE
SYNTAX      INTEGER {
    other          (1),
    local          (2),
    remote        (3)
}
MAX-ACCESS read-create
STATUS      current
DESCRIPTION

```

"The location of the resource (or a collection of resources using any char/wildcard characters) of this conceptual row is either (1) local - the resource is reachable via an interface, or (2) remote - the resource is reachable via a partner DLSw node (or a set of partner DLSw nodes)."

```

DEFVAL { local }
 ::= { dlswDirNBEntry 5 }

```

```

dlswDirNBLocation OBJECT-TYPE
SYNTAX      RowPointer
MAX-ACCESS read-create
STATUS      current
DESCRIPTION

```

"Points to either the ifEntry, dlswTConnConfigEntry, dlswTConnOperEntry, 0.0, or something that is implementation specific. It identifies the location of the NetBIOS name or the set of NetBIOS names."

DEFVAL { null }  
 ::= { dlswDirNBEntry 6 }

dlswDirNBStatus OBJECT-TYPE

SYNTAX INTEGER {  
     unknown                    (1),  
     reachable                  (2),  
     notReachable              (3)  
 }

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object specifies whether DLSw currently believes the NetBIOS name to be accessible at the specified location. The value 'notReachable' allows a configured resource definition to be taken out of service when a search to that resource fails (avoiding a repeat of the search)."

DEFVAL { unknown }  
 ::= { dlswDirNBEntry 7 }

dlswDirNBLFSize OBJECT-TYPE

SYNTAX LFSize

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The largest size of the MAC INFO field (LLC header and data) that a circuit to the NB name can carry through this path."

DEFVAL { lfs65535 }  
 ::= { dlswDirNBEntry 8 }

dlswDirNBRowStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object is used by manager to create or delete the row entry in the dlswDirNBTable following the RowStatus textual convention."

::= { dlswDirNBEntry 9 }

-----  
 -- Resource Locations  
 -----

```

dlswDirLocate          OBJECT IDENTIFIER ::= { dlswDirectory 3 }

-- .....
-- Locate Entries in the dlswDirMacTable for a given MAC address
-- .....
dlswDirLocateMacTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF DlswDirLocateMacEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This table is used to retrieve all entries in the
         dlswDirMacTable that match a given MAC address,
         in the order of the best matched first, the
         second best matched second, and so on, till
         no more entries match the given MAC address."
    ::= { dlswDirLocate 1 }

dlswDirLocateMacEntry OBJECT-TYPE
    SYNTAX      DlswDirLocateMacEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Indexed by dlswDirLocateMacMac and dlswDirLocateMacMatch.
         The first object is the MAC address of interest, and
         the second object is the order in the list of all
         entries that match the MAC address."
    INDEX      { dlswDirLocateMacMac, dlswDirLocateMacMatch }
    ::= { dlswDirLocateMacTable 1 }

DlswDirLocateMacEntry ::= SEQUENCE {
    dlswDirLocateMacMac          MacAddressNC,
    dlswDirLocateMacMatch       INTEGER,
    dlswDirLocateMacLocation    RowPointer
}

dlswDirLocateMacMac OBJECT-TYPE
    SYNTAX      MacAddressNC
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The MAC address to be located."
    ::= { dlswDirLocateMacEntry 1 }

dlswDirLocateMacMatch OBJECT-TYPE
    SYNTAX      INTEGER (1..255)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION

```

"The order of the entries of dlswDirMacTable that match dlswDirLocateMacMac. A value of one represents the entry that best matches the MAC address. A value of two represents the second best matched entry, and so on."

```
::= { dlswDirLocateMacEntry 2 }
```

```
dlswDirLocateMacLocation OBJECT-TYPE
```

```
SYNTAX      RowPointer
```

```
MAX-ACCESS  read-only
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"Points to the dlswDirMacEntry."
```

```
::= { dlswDirLocateMacEntry 3 }
```

```
-- .....
-- Locate Entries in the dlswDirNBTable for a given NetBIOS name
-- .....
```

```
dlswDirLocateNBTable OBJECT-TYPE
```

```
SYNTAX      SEQUENCE OF DlswDirLocateNBEntry
```

```
MAX-ACCESS  not-accessible
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"This table is used to retrieve all entries in the dlswDirNBTable that match a given NetBIOS name, in the order of the best matched first, the second best matched second, and so on, till no more entries match the given NetBIOS name."
```

```
::= { dlswDirLocate 2 }
```

```
dlswDirLocateNBEntry OBJECT-TYPE
```

```
SYNTAX      DlswDirLocateNBEntry
```

```
MAX-ACCESS  not-accessible
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"Indexed by dlswDirLocateNBName and dlswDirLocateNBMatch. The first object is the NetBIOS name of interest, and the second object is the order in the list of all entries that match the NetBIOS name."
```

```
INDEX      { dlswDirLocateNBName, dlswDirLocateNBMatch }
```

```
::= { dlswDirLocateNBTable 1 }
```

```
DlswDirLocateNBEntry ::= SEQUENCE {
```

```
  dlswDirLocateNBName      NBName,
```

```
  dlswDirLocateNBMatch     INTEGER,
```

```
  dlswDirLocateNBLocation  RowPointer
```

```
}
```

```

dlswDirLocateNBName OBJECT-TYPE
    SYNTAX      NBName
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The NetBIOS name to be located (no any char or wildcards)."
```

::= { dlswDirLocateNBEntry 1 }

```

dlswDirLocateNBMatch OBJECT-TYPE
    SYNTAX      INTEGER (1..255)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The order of the entries of dlswDirNBTable
         that match dlswDirLocateNBName. A value of
         one represents the entry that best matches the
         NetBIOS name. A value of two represents the second
         best matched entry, and so on."
```

::= { dlswDirLocateNBEntry 2 }

```

dlswDirLocateNBLocation OBJECT-TYPE
    SYNTAX      RowPointer
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Points to the dlswDirNBEntry."
```

::= { dlswDirLocateNBEntry 3 }

```

-- *****
-- CIRCUIT
-- A circuit is the end-to-end association of two DLSw entities
-- through one or two DLSw nodes. It is the concatenation of
-- two "data links", optionally with an intervening transport
-- connection. The origin of the circuit is the end station that
-- initiates the circuit. The target of the circuit is the end
-- station that receives the initiation.
-- *****
```

```

-----
-- Statistics Related to Circuits
-----
```

```

dlswCircuitStat          OBJECT IDENTIFIER ::= { dlswCircuit 1 }
```

```

dlswCircuitStatActives  OBJECT-TYPE
    SYNTAX      Gauge32
    MAX-ACCESS  read-only
    STATUS      current
```

DESCRIPTION

"The current number of circuits in dlswCircuitTable that are not in the disconnected state."

::= { dlswCircuitStat 1 }

dlswCircuitStatCreates OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of entries ever added to dlswCircuitTable, or reactivated upon exiting 'disconnected' state."

::= { dlswCircuitStat 2 }

-----  
 -- Circuit Table

-- This table is the DLSw entity's view of circuits. There will be a conceptual row in the table associated with each data link.

-- The chart below lists the various possible combinations of origin and target MAC locations and the number of entries in this Circuit Table:

number of entries in the Circuit Table		Origin End Station Location		
		internal	local	remote
Target	internal	NA	2	1
End	local	2	2	1
Station	remote	1	1	NA
Location				

NA: Not applicable

Note:

- (a) IfIndex and RouteInfo are applied only if location is local.
- (b) TDomain and TAddr are applied only if location is remote.

-- Most of statistics related to circuits can be collected from LLC-2 Link Station Table.  
 -----

dlswCircuitTable OBJECT-TYPE

SYNTAX SEQUENCE OF DlswCircuitEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table is the circuit representation in the DLSw entity. Virtual data links are used to represent any internal end stations. There is a conceptual row associated with each data link. Thus, for circuits without an intervening transport connection, there are two conceptual rows for each circuit.

The table consists of the circuits being established, established, and as an implementation option, circuits that have been disconnected. For circuits carried over transport connections, an entry is created after the CUR\_cs was sent or received. For circuits between two locally attached devices, or internal virtual MAC addresses, an entry is created when the equivalent of CUR\_cs sent/received status is reached.

End station 1 (S1) and End station 2 (S2) are used to represent the two end stations of the circuit. S1 is always an end station which is locally attached. S2 may be locally attached or remote. If it is locally attached, the circuit will be represented by two rows indexed by (A, B) and (B, A) where A & B are the relevant MACs/SAPs.

The table may be used to store the causes of disconnection of circuits. It is recommended that the oldest disconnected circuit entry be removed from this table when the memory space of disconnected circuits is needed."

```
::= { dlswCircuit 2 }
```

```
dlswCircuitEntry OBJECT-TYPE
SYNTAX      DlswCircuitEntry
MAX-ACCESS not-accessible
STATUS      current
DESCRIPTION
    ""
INDEX       { dlswCircuitS1Mac,
              dlswCircuitS1Sap,
              dlswCircuitS2Mac,
              dlswCircuitS2Sap }
 ::= { dlswCircuitTable 1 }
```

```
DlswCircuitEntry ::= SEQUENCE {
    dlswCircuitS1Mac      MacAddressNC,
    dlswCircuitS1Sap     OCTET STRING,
    dlswCircuitS1IfIndex INTEGER,
    dlswCircuitS1DlcType DlcType,
    dlswCircuitS1RouteInfo OCTET STRING,
    dlswCircuitS1CircuitId OCTET STRING,
```

```

dlswCircuitS1Dlc          RowPointer,

dlswCircuitS2Mac          MacAddressNC,
dlswCircuitS2Sap          OCTET STRING,
dlswCircuitS2Location     EndStationLocation,
dlswCircuitS2TDomain      OBJECT IDENTIFIER,
dlswCircuitS2TAddress     TAddress,
dlswCircuitS2CircuitId    OCTET STRING,

dlswCircuitOrigin         INTEGER,

dlswCircuitEntryTime      TimeTicks,
dlswCircuitStateTime     TimeTicks,
dlswCircuitState         INTEGER,

dlswCircuitPriority       INTEGER,

dlswCircuitFCSendGrantedUnits  INTEGER,
dlswCircuitFCSendCurrentWndw   INTEGER,
dlswCircuitFCRecvGrantedUnits  INTEGER,
dlswCircuitFCRecvCurrentWndw   INTEGER,
dlswCircuitFCLargestRecvGranted Gauge32,
dlswCircuitFCLargestSendGranted Gauge32,
dlswCircuitFCHalveWndwSents    Counter32,
dlswCircuitFCResetOpSents     Counter32,
dlswCircuitFCHalveWndwRcvds   Counter32,
dlswCircuitFCResetOpRcvds     Counter32,
dlswCircuitDiscReasonLocal    INTEGER,
dlswCircuitDiscReasonRemote   INTEGER,
dlswCircuitDiscReasonRemoteData OCTET STRING
}

-- .....
-- Information related to the End Station 1 (S1).
-- .....
dlswCircuitS1Mac OBJECT-TYPE
    SYNTAX      MacAddressNC
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The MAC Address of End Station 1 (S1) used for this circuit."
        ::= { dlswCircuitEntry 1 }

dlswCircuitS1Sap OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(1))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION

```

"The SAP at End Station 1 (S1) used for this circuit."  
 ::= { dlswCircuitEntry 2 }

dlswCircuitS1IfIndex OBJECT-TYPE  
SYNTAX INTEGER (0..2147483647)  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
"The ifEntry index of the local interface through which S1  
can be reached."  
 ::= { dlswCircuitEntry 3 }

dlswCircuitS1DlcType OBJECT-TYPE  
SYNTAX DlcType  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
"The DLC protocol in use between the DLSw node and S1."  
 ::= { dlswCircuitEntry 4 }

dlswCircuitS1RouteInfo OBJECT-TYPE  
SYNTAX OCTET STRING (SIZE (0..30))  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
"If source-route bridging is in use between the DLSw  
node and S1, this is the routing information field  
describing the path between the two devices.  
Otherwise the value will be an OCTET STRING of  
zero length."  
 ::= { dlswCircuitEntry 5 }

dlswCircuitS1CircuitId OBJECT-TYPE  
SYNTAX OCTET STRING (SIZE (0 | 8))  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
"The Circuit ID assigned by this DLSw node to this circuit.  
The first four octets are the DLC port Id, and  
the second four octets are the Data Link Correlator.  
If the DLSw SSP was not used to establish this circuit,  
the value will be a string of zero length."  
 ::= { dlswCircuitEntry 6 }

dlswCircuitS1Dlc OBJECT-TYPE  
SYNTAX RowPointer  
MAX-ACCESS read-only  
STATUS current

## DESCRIPTION

"Points to a conceptual row of the underlying DLC MIB, which could either be the standard MIBs (e.g., the SDLC), or an enterprise-specific DLC MIB."

::= { dlswCircuitEntry 7 }

```
-- .....
-- Information related to the End Station 2 (S2).
-- .....
```

dlswCircuitsS2Mac OBJECT-TYPE

SYNTAX MacAddressNC

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"The MAC Address of End Station 2 (S2) used for this circuit."

::= { dlswCircuitEntry 8 }

dlswCircuitsS2Sap OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(1))

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"The SAP at End Station 2 (S2) used for this circuit."

::= { dlswCircuitEntry 9 }

dlswCircuitsS2Location OBJECT-TYPE

SYNTAX EndStationLocation

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The location of End Station 2 (S2).

If the location of End Station 2 is local, the interface information will be available in the conceptual row whose S1 and S2 are the S2 and the S1 of this conceptual row, respectively."

::= { dlswCircuitEntry 10 }

dlswCircuitsS2TDomain OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"If the location of End Station 2 is remote, this value is the transport domain of the transport protocol the circuit is running over. Otherwise, the value is 0.0."

::= { dlswCircuitEntry 11 }

```

dlswCircuitS2TAddress OBJECT-TYPE
    SYNTAX      TAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "If the location of End Station 2 is remote,
         this object contains the address of the partner
         DLSw, else it will be an OCTET STRING of zero length."
    ::= { dlswCircuitEntry 12 }

dlswCircuitS2CircuitId OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE (0 | 8))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The Circuit ID assigned to this circuit by the partner
         DLSw node. The first four octets are the DLC port Id, and
         the second four octets are the Data Link Correlator.
         If the DLSw SSP was not used to establish this circuit,
         the value will be a string of zero length."
    ::= { dlswCircuitEntry 13 }

-- .....
dlswCircuitOrigin OBJECT-TYPE
    SYNTAX      INTEGER {
        s1      (1),
        s2      (2)
    }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object specifies which of the two end stations
         initiated the establishment of this circuit."
    ::= { dlswCircuitEntry 14 }

-- .....
-- Operational information related to this circuit.
-- .....
dlswCircuitEntryTime OBJECT-TYPE
    SYNTAX      TimeTicks
    UNITS       "hundredths of a second"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The amount of time (in hundredths of a second) since this
         circuit table conceptual row was created."
    ::= { dlswCircuitEntry 15 }

```

```

dlswCircuitStateTime OBJECT-TYPE
    SYNTAX      TimeTicks
    UNITS       "hundredths of a second"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The amount of time (in hundredths of a second) since this
         circuit entered the current state."
    ::= { dlswCircuitEntry 16 }

```

```

dlswCircuitState OBJECT-TYPE
    SYNTAX      INTEGER {
        disconnected          (1),
        circuitStart         (2),
        resolvePending       (3),
        circuitPending       (4),
        circuitEstablished   (5),
        connectPending       (6),
        contactPending       (7),
        connected            (8),
        disconnectPending    (9),
        haltPending          (10),
        haltPendingNoack     (11),
        circuitRestart       (12),
        restartPending       (13)
    }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The current state of this circuit. The agent, implementation
         specific, may choose to keep entries for some period of time
         after circuit disconnect, so the manager can gather the time
         and cause of disconnection.

         While all of the specified values may be returned from a GET
         operation, the only SETable value is 'disconnectPending'.
         When this value is set, DLSw should perform the appropriate
         action given its previous state (e.g., send HALT_DL if the
         state was 'connected') to bring the circuit down to the
         'disconnected' state. Both the partner DLSw and local end
         station(s) should be notified as appropriate.

         This MIB provides no facility to re-establish a disconnected
         circuit, because in DLSw this should be an end station-driven
         function."
    ::= { dlswCircuitEntry 17 }

```

```

dlswCircuitPriority OBJECT-TYPE

```

```

SYNTAX      INTEGER {
  unsupported (1),
  low         (2),
  medium      (3),
  high        (4),
  highest     (5)
}

```

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The transmission priority of this circuit as understood by this DLSw node. This value is determined by the two DLSw nodes at circuit startup time. If this DLSw node does not support DLSw circuit priority, the value 'unsupported' should be returned."

::= { dlswCircuitEntry 18 }

```

-- .....
-- Pacing Objects:
-- These objects are applicable if DLSw is using the SSP circuit
-- pacing protocol to control the flow between the two data links
-- in this circuit.
-- .....

```

dlswCircuitFCSendGrantedUnits OBJECT-TYPE

SYNTAX INTEGER (0..65535)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of paced SSP messages that this DLSw is currently authorized to send on this circuit before it must stop and wait for an additional flow control indication from the partner DLSw.

The value zero should be returned if this circuit is not running the DLSw pacing protocol."

::= { dlswCircuitEntry 19 }

dlswCircuitFCSendCurrentWndw OBJECT-TYPE

SYNTAX INTEGER (0..65535)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The current window size that this DLSw is using in its role as a data sender. This is the value by which this DLSw would increase the number of messages it is authorized to send, if it were to receive a flow control indication with the bits specifying 'repeat window'.

The value zero should be returned if this circuit is not running the DLSw pacing protocol."

::= { dlswCircuitEntry 20 }

dlswCircuitFCRecvGrantedUnits OBJECT-TYPE

SYNTAX INTEGER (0..65535)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The current number of paced SSP messages that this DLSw has authorized the partner DLSw to send on this circuit before the partner DLSw must stop and wait for an additional flow control indication from this DLSw.

The value zero should be returned if this circuit is not running the DLSw pacing protocol."

::= { dlswCircuitEntry 21 }

dlswCircuitFCRecvCurrentWndw OBJECT-TYPE

SYNTAX INTEGER (0..65535)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The current window size that this DLSw is using in its role as a data receiver. This is the number of additional paced SSP messages that this DLSw would be authorizing its DLSw partner to send, if this DLSw were to send a flow control indication with the bits specifying 'repeat window'.

The value zero should be returned if this circuit is not running the DLSw pacing protocol."

::= { dlswCircuitEntry 22 }

dlswCircuitFCLargestRecvGranted OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The largest receive window size granted by this DLSw during the current activation of this circuit. This is not the largest number of messages granted at any time, but the largest window size as represented by FCIND operator bits.

The value zero should be returned if this circuit is not running the DLSw pacing protocol."

::= { dlswCircuitEntry 23 }

dlswCircuitFCLargestSendGranted OBJECT-TYPE

```
SYNTAX      Gauge32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "The largest send (with respect to this DLSw) window size
    granted by the partner DLSw during the current activation of
    this circuit.
```

```
    The value zero should be returned if this circuit is not
    running the DLSw pacing protocol."
```

```
::= { dlswCircuitEntry 24 }
```

```
dlswCircuitFCHalveWndwSents  OBJECT-TYPE
```

```
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
```

```
    "The number of Halve Window operations this DLSw has sent on
    this circuit, in its role as a data receiver.
```

```
    The value zero should be returned if this circuit is not
    running the DLSw pacing protocol."
```

```
::= { dlswCircuitEntry 25 }
```

```
dlswCircuitFCResetOpSents  OBJECT-TYPE
```

```
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
```

```
    "The number of Reset Window operations this DLSw has sent on
    this circuit, in its role as a data receiver.
```

```
    The value zero should be returned if this circuit is not
    running the DLSw pacing protocol."
```

```
::= { dlswCircuitEntry 26 }
```

```
dlswCircuitFCHalveWndwRcvds  OBJECT-TYPE
```

```
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
```

```
    "The number of Halve Window operations this DLSw has received on
    this circuit, in its role as a data sender.
```

```
    The value zero should be returned if this circuit is not
    running the DLSw pacing protocol."
```

```
::= { dlswCircuitEntry 27 }
```

```

dlswCircuitFCResetOpRcvds OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of Reset Window operations this DLSw has received on
        this circuit, in its role as a data sender.
        The value zero should be returned if this circuit is not
        running the DLSw pacing protocol."
    ::= { dlswCircuitEntry 28 }

```

```

-- .....
-- Information about the circuit disconnection
-- .....

```

```

dlswCircuitDiscReasonLocal OBJECT-TYPE
    SYNTAX      INTEGER {
        endStationDiscRcvd          (1),
        endStationDlcError          (2),
        protocolError                (3),
        operatorCommand              (4),
        haltDlRcvd                   (5),
        haltDlNoAckRcvd              (6),
        transportConnClosed          (7)
    }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The reason why this circuit was last disconnected, as seen
        by this DLSw node.

        This object is present only if the agent keeps circuit
        table entries around for some period after circuit disconnect."
    ::= { dlswCircuitEntry 29 }

```

```

dlswCircuitDiscReasonRemote OBJECT-TYPE
    SYNTAX      INTEGER {
        unknown                       (1),
        endStationDiscRcvd            (2),
        endStationDlcError            (3),
        protocolError                  (4),
        operatorCommand                (5)
    }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The generic reason code why this circuit was last
        disconnected, as reported by the DLSw partner in a HALT_DL

```

or HALT\_DL\_NOACK. If the partner does not send a reason code in these messages, or the DLSw implementation does not report receiving one, the value 'unknown' is returned.

This object is present only if the agent keeps circuit table entries around for some period after circuit disconnect."

::= { dlswCircuitEntry 30 }

dlswCircuitDiscReasonRemoteData OBJECT-TYPE

SYNTAX OCTET STRING (SIZE (0 | 4))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Implementation-specific data reported by the DLSw partner in a HALT\_DL or HALT\_DL\_NOACK, to help specify how and why this circuit was last disconnected. If the partner does not send this data in these messages, or the DLSw implementation does not report receiving it, a string of zero length is returned.

This object is present only if the agent keeps circuit table entries around for some period after circuit disconnect."

::= { dlswCircuitEntry 31 }

-- .....
-- Statistics related to this circuit.
-- All statistics are in LLC-2 Link Station Statistical Table.
-- All SDLC statistics are in SDLC MIB
-- .....

-- \*\*\*\*\*
-- DLSW SDLC EXTENSION
-- \*\*\*\*\*

dlswSdlcLsEntries OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of entries in dlswSdlcLsTable."

::= { dlswSdlc 1 }

-- .....
dlswSdlcLsTable OBJECT-TYPE
SYNTAX SEQUENCE OF DlswSdlcLsEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION

"The table defines the virtual MAC addresses for those  
 SDLC link stations that participate in data link switching."  
 ::= { dlswSdlc 2 }

dlswSdlcLsEntry OBJECT-TYPE  
 SYNTAX DlswSdlcLsEntry  
 MAX-ACCESS not-accessible  
 STATUS current  
 DESCRIPTION  
 "The index of this table is the ifIndex value for the  
 SDLC port which owns this link station and the poll  
 address of the particular SDLC link station."  
 INDEX { ifIndex, sdlcLSAddress }  
 ::= { dlswSdlcLsTable 1 }

DlswSdlcLsEntry ::= SEQUENCE {  
 dlswSdlcLsLocalMac MacAddressNC,  
 dlswSdlcLsLocalSap OCTET STRING,  
 dlswSdlcLsLocalIdBlock DisplayString,  
 dlswSdlcLsLocalIdNum DisplayString,  
 dlswSdlcLsRemoteMac MacAddressNC,  
 dlswSdlcLsRemoteSap OCTET STRING,  
 dlswSdlcLsRowStatus RowStatus  
 }

dlswSdlcLsLocalMac OBJECT-TYPE  
 SYNTAX MacAddressNC  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION  
 "The virtual MAC address used to represent the SDLC-attached  
 link station to the rest of the DLSw network."  
 ::= { dlswSdlcLsEntry 1 }

dlswSdlcLsLocalSap OBJECT-TYPE  
 SYNTAX OCTET STRING (SIZE(1))  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION  
 "The SAP used to represent this link station."  
 ::= { dlswSdlcLsEntry 2 }

dlswSdlcLsLocalIdBlock OBJECT-TYPE  
 SYNTAX DisplayString (SIZE (0 | 3))  
 MAX-ACCESS read-create  
 STATUS current  
 DESCRIPTION  
 "The block number is the first three digits of the node\_id,

if available. These 3 hexadecimal digits identify the product."

```
DEFVAL { 'H }
 ::= { dlswSdlcLsEntry 3 }
```

dlswSdlcLsLocalIdNum OBJECT-TYPE

```
SYNTAX      DisplayString (SIZE (0 | 5))
```

```
MAX-ACCESS  read-create
```

```
STATUS      current
```

```
DESCRIPTION
```

"The ID number is the last 5 digits of the node\_id, if available. These 5 hexadecimal digits are administratively defined and combined with the 3 digit block number form the node\_id. This node\_id is used to identify the local node and is included in SNA XIDs."

```
DEFVAL { 'H }
 ::= { dlswSdlcLsEntry 4 }
```

dlswSdlcLsRemoteMac OBJECT-TYPE

```
SYNTAX      MacAddressNC
```

```
MAX-ACCESS  read-create
```

```
STATUS      current
```

```
DESCRIPTION
```

"The MAC address to which DLSw should attempt to connect this link station. If this information is not available, a length of zero for this object should be returned."

```
DEFVAL { 'H }
 ::= { dlswSdlcLsEntry 5 }
```

dlswSdlcLsRemoteSap OBJECT-TYPE

```
SYNTAX      OCTET STRING (SIZE (0 | 1))
```

```
MAX-ACCESS  read-create
```

```
STATUS      current
```

```
DESCRIPTION
```

"The SAP of the remote station to which this link station should be connected. If this information is not available, a length of zero for this object should be returned."

```
DEFVAL { 'H }
 ::= { dlswSdlcLsEntry 6 }
```

dlswSdlcLsRowStatus OBJECT-TYPE

```
SYNTAX      RowStatus
```

```
MAX-ACCESS  read-create
```

```
STATUS      current
```

```
DESCRIPTION
```

"This object is used by the manager to create or delete the row entry in the dlswSdlcLsTable"

```

    following the RowStatus textual convention."
 ::= { dlswSdlcLsEntry 7 }

```

```

-- *****
-- TRAP GENERATION CONTROL
-- *****
dlswTrapControl OBJECT IDENTIFIER ::= { dlswNode 10}

```

```
dlswTrapCntlTConnPartnerReject OBJECT-TYPE
```

```

SYNTAX      INTEGER {
    enabled      (1),
    disabled     (2),
    partial      (3)
}

```

```
MAX-ACCESS read-write
```

```
STATUS      current
```

```
DESCRIPTION
```

```

"Indicates whether the DLSw is permitted to emit partner
reject related traps.  With the value of 'enabled'
the DLSw will emit all partner reject related traps.
With the value of 'disabled' the DLSw will not emit
any partner reject related traps.  With the value
of 'partial' the DLSw will only emits partner reject
traps for CapEx reject.  The changes take effect
immediately."

```

```
 ::= { dlswTrapControl 1 }

```

```
dlswTrapCntlTConnProtViolation OBJECT-TYPE
```

```
SYNTAX      TruthValue
```

```
MAX-ACCESS read-write
```

```
STATUS      current
```

```
DESCRIPTION
```

```

"Indicates whether the DLSw is permitted to generate
protocol-violation traps on the events such as
window size violation.  The changes take effect
immediately."

```

```
 ::= { dlswTrapControl 2 }

```

```
dlswTrapCntlTConn OBJECT-TYPE
```

```

SYNTAX      INTEGER {
    enabled      (1),
    disabled     (2),
    partial      (3)
}

```

```
MAX-ACCESS read-write
```

```
STATUS      current
```

```
DESCRIPTION
```

"Indicates whether the DLSw is permitted to emit transport connection up and down traps. With the value of 'enabled' the DLSw will emit traps when connections enter 'connected' and 'disconnected' states. With the value of 'disabled' the DLSw will not emit traps when connections enter of 'connected' and 'disconnected' states. With the value of 'partial' the DLSw will only emits transport connection down traps when the connection is closed with busy. The changes take effect immediately."

```
::= { dlswTrapControl 3 }
```

```
dlswTrapCntlCircuit OBJECT-TYPE
```

```
SYNTAX      INTEGER {
    enabled   (1),
    disabled  (2),
    partial   (3)
}
```

```
MAX-ACCESS read-write
```

```
STATUS      current
```

```
DESCRIPTION
```

"Indicates whether the DLSw is permitted to generate circuit up and down traps. With the value of 'enabled' the DLSw will emit traps when circuits enter 'connected' and 'disconnected' states. With the value of 'disabled' the DLSw will not emit traps when circuits enter of 'connected' and 'disconnected' states. With the value of 'partial' the DLSw will emit traps only for those circuits that are initiated by this DLSw, e.g., originating the CUR\_CS message. The changes take effect immediately."

```
::= { dlswTrapControl 4 }
```

```
-- *****
-- NOTIFICATIONS, i.e., TRAP DEFINITIONS
-- *****
```

```
dlswTraps          OBJECT IDENTIFIER ::= { dlswMIB 0 }
```

```
-- -----
-- This section defines the well-known notifications sent by
-- DLSW agents.
-- Care must be taken to insure that no particular notification
-- is sent to a single receiving entity more often than once
-- every five seconds.
--
-- Traps includes:
-- (1) Partner rejected (capEx rejection, not in partner list, etc.)
-- (2) DLSw protocol violation (e.g., window size violation, etc.)
-- (3) Transport connection up/down
```

```
-- (4) Circuit up/down
```

```
-----
```

```
dlswTrapTConnPartnerReject NOTIFICATION-TYPE
  OBJECTS { dlswTConnOperTDomain, dlswTConnOperRemoteTAddr
  }
  STATUS      current
  DESCRIPTION
    "This trap is sent each time a transport connection
     is rejected by a partner DLSw during Capabilities
     Exchanges. The emission of this trap is controlled
     by dlswTrapCntlTConnPartnerReject."
 ::= { dlswTraps 1 }

dlswTrapTConnProtViolation NOTIFICATION-TYPE
  OBJECTS { dlswTConnOperTDomain, dlswTConnOperRemoteTAddr
  }
  STATUS      current
  DESCRIPTION
    "This trap is sent each time a protocol violation is
     detected for a transport connection. The emission of this
     trap is controlled by dlswTrapCntlTConnProtViolation."
 ::= { dlswTraps 2 }

dlswTrapTConnUp NOTIFICATION-TYPE
  OBJECTS { dlswTConnOperTDomain, dlswTConnOperRemoteTAddr
  }
  STATUS      current
  DESCRIPTION
    "This trap is sent each time a transport connection
     enters 'connected' state. The emission of this trap
     is controlled by dlswTrapCntlTConn."
 ::= { dlswTraps 3 }

dlswTrapTConnDown NOTIFICATION-TYPE
  OBJECTS { dlswTConnOperTDomain, dlswTConnOperRemoteTAddr
  }
  STATUS      current
  DESCRIPTION
    "This trap is sent each time a transport connection
     enters 'disconnected' state. The emission of this trap
     is controlled by dlswTrapCntlTConn."
 ::= { dlswTraps 4 }

dlswTrapCircuitUp NOTIFICATION-TYPE
  OBJECTS { dlswCircuitS1Mac, dlswCircuitS1Sap,
           dlswCircuitS2Mac, dlswCircuitS2Sap
```

```

}
STATUS      current
DESCRIPTION
  "This trap is sent each time a circuit enters 'connected'
  state. The emission of this trap is controlled by
  dlswTrapCntlCircuit."
 ::= { dlswTraps 5 }

```

```

dlswTrapCircuitDown NOTIFICATION-TYPE
  OBJECTS { dlswCircuitS1Mac, dlswCircuitS1Sap,
            dlswCircuitS2Mac, dlswCircuitS2Sap
  }
  STATUS      current
  DESCRIPTION
    "This trap is sent each time a circuit enters 'disconnected'
    state. The emission of this trap is controlled by
    dlswTrapCntlCircuit."
 ::= { dlswTraps 6 }

```

```

-- *****
-- CONFORMANCE INFORMATION
-- *****
dlswConformance      OBJECT IDENTIFIER ::= { dlsw 3 }

dlswCompliances      OBJECT IDENTIFIER ::= { dlswConformance 1 }
dlswGroups            OBJECT IDENTIFIER ::= { dlswConformance 2 }

```

```

-----
-- COMPLIANCE STATEMENTS
-----

```

```

-- .....
-- Core compliance for all DLSw entities
-- .....

```

```

dlswCoreCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION
    "The core compliance statement for all DLSw nodes."
  MODULE
    MANDATORY-GROUPS {
      dlswNodeGroup,
      dlswTConnStatGroup,
      dlswTConnConfigGroup,
      dlswTConnOperGroup,
      dlswInterfaceGroup,
      dlswCircuitGroup,
      dlswCircuitStatGroup,

```

dlswNotificationGroup }

GROUP dlswNodeNBGroup

DESCRIPTION

"The DLSw NetBIOS Node group is mandatory only for those DLSw entities that implement NetBIOS."

GROUP dlswTConnNBGroup

DESCRIPTION

"The DLSw NetBIOS Transport Connection group is mandatory only for those DLSw entities that implement NetBIOS."

OBJECT dlswNodeStatus

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT dlswNodeVirtualSegmentLFSize

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT dlswNodeResourceNBExclusivity

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT dlswNodeResourceMacExclusivity

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT dlswTrapCntlTConnPartnerReject

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT dlswTrapCntlTConnProtViolation

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT dlswTrapCntlTConn

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

```
OBJECT dlswTrapCntlCircuit
  MIN-ACCESS read-only
  DESCRIPTION
    "Write access is not required."

OBJECT dlswTConnConfigTDomain
  MIN-ACCESS read-only
  DESCRIPTION
    "Write access is not required."

OBJECT dlswTConnConfigLocalTAddr
  MIN-ACCESS read-only
  DESCRIPTION
    "Write access is not required."

OBJECT dlswTConnConfigRemoteTAddr
  MIN-ACCESS read-only
  DESCRIPTION
    "Write access is not required."

OBJECT dlswTConnConfigEntryType
  MIN-ACCESS read-only
  DESCRIPTION
    "Write access is not required."

OBJECT dlswTConnConfigGroupDefinition
  MIN-ACCESS read-only
  DESCRIPTION
    "Write access is not required."

OBJECT dlswTConnConfigSetupType
  MIN-ACCESS read-only
  DESCRIPTION
    "Write access is not required."

OBJECT dlswTConnConfigSapList
  MIN-ACCESS read-only
  DESCRIPTION
    "Write access is not required."

OBJECT dlswTConnConfigAdvertiseMacNB
  MIN-ACCESS read-only
  DESCRIPTION
    "Write access is not required."

OBJECT dlswTConnConfigInitCirRecvWdw
  MIN-ACCESS read-only
  DESCRIPTION
```

"Write access is not required."

OBJECT dlswTConnConfigRowStatus  
 MIN-ACCESS read-only  
 DESCRIPTION  
 "Write access is not required."

OBJECT dlswTConnOperState  
 MIN-ACCESS read-only  
 DESCRIPTION  
 "Write access is not required."

OBJECT dlswIfRowStatus  
 MIN-ACCESS read-only  
 DESCRIPTION  
 "Write access is not required."

OBJECT dlswIfVirtualSegment  
 MIN-ACCESS read-only  
 DESCRIPTION  
 "Write access is not required."

OBJECT dlswIfSapList  
 MIN-ACCESS read-only  
 DESCRIPTION  
 "Write access is not required."

OBJECT dlswCircuitState  
 MIN-ACCESS read-only  
 DESCRIPTION  
 "Write access is not required."

::= { dlswCompliances 1 }

-- .....  
 -- Compliance for all DLSw entities that provide TCP transport.  
 -- .....

dlswTConnTcpCompliance MODULE-COMPLIANCE  
 STATUS current  
 DESCRIPTION  
 "Compliance for DLSw nodes that use TCP as a  
 transport connection protocol."  
 MODULE  
 MANDATORY-GROUPS {  
 dlswTConnTcpConfigGroup,  
 dlswTConnTcpOperGroup }  
 OBJECT dlswTConnTcpConfigKeepAliveInt

```

    MIN-ACCESS read-only
    DESCRIPTION
        "Write access is not required."

OBJECT dlswTConnTcpConfigTcpConnections
    MIN-ACCESS read-only
    DESCRIPTION
        "Write access is not required."

OBJECT dlswTConnTcpConfigMaxSegmentSize
    MIN-ACCESS read-only
    DESCRIPTION
        "Write access is not required."

 ::= { dlswCompliances 2 }

-- .....
-- Compliance for all DLSw Entities that implement a directory
-- .....
dlswDirCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
        "Compliance for DLSw nodes that provide a directory
        function."
    MODULE
        MANDATORY-GROUPS {
            dlswDirGroup }

    GROUP dlswDirNBGroup
        DESCRIPTION
            "The DLSw NetBIOS group is mandatory only for
            those DLSw entities that implement NetBIOS."

    OBJECT dlswDirMacMac
        MIN-ACCESS read-only
        DESCRIPTION
            "Write access is not required."

    OBJECT dlswDirMacMask
        MIN-ACCESS read-only
        DESCRIPTION
            "Write access is not required."

    OBJECT dlswDirMacEntryType
        MIN-ACCESS read-only
        DESCRIPTION
            "Write access is not required."

```

OBJECT dlswDirMacLocationType  
MIN-ACCESS read-only  
DESCRIPTION  
"Write access is not required."

OBJECT dlswDirMacLocation  
MIN-ACCESS read-only  
DESCRIPTION  
"Write access is not required."

OBJECT dlswDirMacStatus  
MIN-ACCESS read-only  
DESCRIPTION  
"Write access is not required."

OBJECT dlswDirMacLFSize  
MIN-ACCESS read-only  
DESCRIPTION  
"Write access is not required."

OBJECT dlswDirMacRowStatus  
MIN-ACCESS read-only  
DESCRIPTION  
"Write access is not required."

OBJECT dlswDirNBName  
MIN-ACCESS read-only  
DESCRIPTION  
"Write access is not required."

OBJECT dlswDirNBNameType  
MIN-ACCESS read-only  
DESCRIPTION  
"Write access is not required."

OBJECT dlswDirNBEntryType  
MIN-ACCESS read-only  
DESCRIPTION  
"Write access is not required."

OBJECT dlswDirNBLocationType  
MIN-ACCESS read-only  
DESCRIPTION  
"Write access is not required."

OBJECT dlswDirNBLocation  
MIN-ACCESS read-only  
DESCRIPTION

"Write access is not required."

```
OBJECT dlswDirNBStatus
MIN-ACCESS read-only
DESCRIPTION
    "Write access is not required."
```

```
OBJECT dlswDirNBLFSize
MIN-ACCESS read-only
DESCRIPTION
    "Write access is not required."
```

```
OBJECT dlswDirNBRowStatus
MIN-ACCESS read-only
DESCRIPTION
    "Write access is not required."
```

```
::= { dlswCompliances 3 }
```

```
-- .....
-- Compliance for all DLSw entities that provide an ordered
-- list of directory entries that match a resource
-- .....
```

```
dlswDirLocateCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
    "Compliance for DLSw nodes that provide an ordered
    list of directory entries for a given resource."
MODULE
    MANDATORY-GROUPS {
        dlswDirLocateGroup }

    GROUP dlswDirLocateNBGroup
    DESCRIPTION
        "The DLSw NetBIOS group is mandatory only for
        those DLSw entities that implement NetBIOS."
```

```
::= { dlswCompliances 4 }
```

```
-- .....
-- Compliance for all DLSw entities that support SDLC end stations
-- .....
```

```
dlswSdlcCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
    "Compliance for DLSw nodes that support SDLC."
MODULE
    MANDATORY-GROUPS {
```

dlswSdlcGroup }

OBJECT dlswSdlcLsLocalMac
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT dlswSdlcLsLocalSap
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT dlswSdlcLsLocalIdBlock
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT dlswSdlcLsLocalIdNum
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT dlswSdlcLsRemoteMac
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT dlswSdlcLsRemoteSap
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT dlswSdlcLsRowStatus
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

::= { dlswCompliances 5 }

-- -----
-- CONFORMANCE GROUPS
-- -----

-- .....
-- Node Conformance Group
-- .....

dlswNodeGroup OBJECT-GROUP
OBJECTS {

```

    dlswNodeVersion,
    dlswNodeVendorID,
    dlswNodeVersionString,
    dlswNodeStdPacingSupport,
    dlswNodeStatus,
    dlswNodeUpTime,
    dlswNodeVirtualSegmentLFSize,
    dlswNodeResourceMacExclusivity,
    dlswTrapCntlTConnPartnerReject,
    dlswTrapCntlTConnProtViolation,
    dlswTrapCntlTConn,
    dlswTrapCntlCircuit
  }
STATUS current
DESCRIPTION
  "Conformance group for DLSw node general information."
 ::= { dlswGroups 1 }

-- .....
dlswNodeNBGroup OBJECT-GROUP
OBJECTS {
  dlswNodeResourceNBExclusivity
}
STATUS current
DESCRIPTION
  "Conformance group for DLSw node general information
  specifically for nodes that support NetBIOS."
 ::= { dlswGroups 2 }

-- .....
dlswTConnStatGroup OBJECT-GROUP
OBJECTS {
  dlswTConnStatActiveConnections,
  dlswTConnStatCloseIdles,
  dlswTConnStatCloseBusys
}
STATUS current
DESCRIPTION
  "Conformance group for statistics for transport
  connections."
 ::= { dlswGroups 3 }

-- .....
dlswTConnConfigGroup OBJECT-GROUP
OBJECTS {
  dlswTConnConfigTDomain,
  dlswTConnConfigLocalTAddr,
  dlswTConnConfigRemoteTAddr,

```

```

    dlswTConnConfigLastModifyTime,
    dlswTConnConfigEntryType,
    dlswTConnConfigGroupDefinition,
    dlswTConnConfigSetupType,
    dlswTConnConfigSapList,
    dlswTConnConfigAdvertiseMacNB,
    dlswTConnConfigInitCirRecvWndw,
    dlswTConnConfigOpens,
    dlswTConnConfigRowStatus
    }
STATUS current
DESCRIPTION
    "Conformance group for the configuration of
    transport connections."
 ::= { dlswGroups 4 }

-- .....
dlswTConnOperGroup OBJECT-GROUP
OBJECTS {
    dlswTConnOperLocalTAddr,
    dlswTConnOperEntryTime,
    dlswTConnOperConnectTime,
    dlswTConnOperState,
    dlswTConnOperConfigIndex,
    dlswTConnOperFlowCntlMode,
    dlswTConnOperPartnerVersion,
    dlswTConnOperPartnerVendorID,
    dlswTConnOperPartnerVersionStr,
    dlswTConnOperPartnerInitPacingWndw,
    dlswTConnOperPartnerSapList,
    dlswTConnOperPartnerMacExcl,
    dlswTConnOperPartnerMacInfo,
    dlswTConnOperDiscTime,
    dlswTConnOperDiscReason,
    dlswTConnOperDiscActiveCir,
    dlswTConnOperInDataPkts,
    dlswTConnOperOutDataPkts,
    dlswTConnOperInDataOctets,
    dlswTConnOperOutDataOctets,
    dlswTConnOperInCntlPkts,
    dlswTConnOperOutCntlPkts,
    dlswTConnOperCURExSents,
    dlswTConnOperICRexRcvds,
    dlswTConnOperCURExRcvds,
    dlswTConnOperICRexSents,
    dlswTConnOperCirCreates,
    dlswTConnOperCircuits
    }

```

```

STATUS current
DESCRIPTION
    "Conformance group for operation information for
    transport connections."
 ::= { dlswGroups 5 }
-- .....
dlswTConnNBGroup OBJECT-GROUP
OBJECTS {
    dlswTConnOperPartnerNBExcl,
    dlswTConnOperPartnerNBInfo,
    dlswTConnOperNQexSents,
    dlswTConnOperNRexRcvds,
    dlswTConnOperNQexRcvds,
    dlswTConnOperNRexSents
}
STATUS current
DESCRIPTION
    "Conformance group for operation information for
    transport connections, specifically for nodes
    that support NetBIOS."
 ::= { dlswGroups 6 }
-- .....
dlswTConnTcpConfigGroup OBJECT-GROUP
OBJECTS {
    dlswTConnTcpConfigKeepAliveInt,
    dlswTConnTcpConfigTcpConnections,
    dlswTConnTcpConfigMaxSegmentSize
}
STATUS current
DESCRIPTION
    "Conformance group for configuration information for
    transport connections using TCP."
 ::= { dlswGroups 7 }
-- .....
dlswTConnTcpOperGroup OBJECT-GROUP
OBJECTS {
    dlswTConnTcpOperKeepAliveInt,
    dlswTConnTcpOperPrefTcpConnections,
    dlswTConnTcpOperTcpConnections
}
STATUS current
DESCRIPTION
    "Conformance group for operation information for
    transport connections using TCP."
 ::= { dlswGroups 8 }

```

```

-- .....
dlswInterfaceGroup OBJECT-GROUP
  OBJECTS {
    dlswIfRowStatus,
    dlswIfVirtualSegment,
    dlswIfSapList
  }
  STATUS current
  DESCRIPTION
    "Conformance group for DLsw interfaces."
    ::= { dlswGroups 9 }

-- .....
dlswDirGroup OBJECT-GROUP
  OBJECTS {
    dlswDirMacEntries,
    dlswDirMacCacheHits,
    dlswDirMacCacheMisses,
    dlswDirMacCacheNextIndex,
    dlswDirMacMac,
    dlswDirMacMask,
    dlswDirMacEntryType,
    dlswDirMacLocationType,
    dlswDirMacLocation,
    dlswDirMacStatus,
    dlswDirMacLFSize,
    dlswDirMacRowStatus
  }
  STATUS current
  DESCRIPTION
    "Conformance group for DLsw directory using MAC
    addresses."
    ::= { dlswGroups 10 }

-- .....
dlswDirNBGroup OBJECT-GROUP
  OBJECTS {
    dlswDirNBEntries,
    dlswDirNBCacheHits,
    dlswDirNBCacheMisses,
    dlswDirNBCacheNextIndex,
    dlswDirNBName,
    dlswDirNBNameType,
    dlswDirNBEntryType,
    dlswDirNBLocationType,
    dlswDirNBLocation,
    dlswDirNBStatus,
    dlswDirNBLFSize,

```

```

        dlswDirNBRowStatus
        }
    STATUS current
    DESCRIPTION
        "Conformance group for DLSw directory using NetBIOS
        names."
    ::= { dlswGroups 11 }

-- .....
dlswDirLocateGroup  OBJECT-GROUP
    OBJECTS {
        dlswDirLocateMacLocation
    }
    STATUS current
    DESCRIPTION
        "Conformance group for a node that can return directory
        entry order for a given MAC address."
    ::= { dlswGroups 12 }

-- .....
dlswDirLocateNBGroup  OBJECT-GROUP
    OBJECTS {
        dlswDirLocateNBLocation
    }
    STATUS current
    DESCRIPTION
        "Conformance group for a node that can return directory
        entry order for a given NetBIOS name."
    ::= { dlswGroups 13 }

-- .....
dlswCircuitStatGroup  OBJECT-GROUP
    OBJECTS {
        dlswCircuitStatActives,
        dlswCircuitStatCreates
    }
    STATUS current
    DESCRIPTION
        "Conformance group for statistics about circuits."
    ::= { dlswGroups 14 }

-- .....
dlswCircuitGroup  OBJECT-GROUP
    OBJECTS {
        dlswCircuitS1IfIndex,
        dlswCircuitS1DlcType,
        dlswCircuitS1RouteInfo,
        dlswCircuitS1CircuitId,

```

```

    dlswCircuitS1Dlc,
    dlswCircuits2Location,
    dlswCircuits2TDomain,
    dlswCircuits2TAddress,
    dlswCircuits2CircuitId,
    dlswCircuitOrigin,
    dlswCircuitEntryTime,
    dlswCircuitStateTime,
    dlswCircuitState,
    dlswCircuitPriority,
    dlswCircuitFCSendGrantedUnits,
    dlswCircuitFCSendCurrentWndw,
    dlswCircuitFCRecvGrantedUnits,
    dlswCircuitFCRecvCurrentWndw,
    dlswCircuitFCLargestRecvGranted,
    dlswCircuitFCLargestSendGranted,
    dlswCircuitFCHalveWndwSents,
    dlswCircuitFCResetOpSents,
    dlswCircuitFCHalveWndwRcvds,
    dlswCircuitFCResetOpRcvds,
    dlswCircuitDiscReasonLocal,
    dlswCircuitDiscReasonRemote,
    dlswCircuitDiscReasonRemoteData
  }
STATUS current
DESCRIPTION
  "Conformance group for DLSw circuits."
 ::= { dlswGroups 15 }

-- .....
dlswSdlcGroup OBJECT-GROUP
  OBJECTS {
    dlswSdlcLsEntries,
    dlswSdlcLsLocalMac,
    dlswSdlcLsLocalSap,
    dlswSdlcLsLocalIdBlock,
    dlswSdlcLsLocalIdNum,
    dlswSdlcLsRemoteMac,
    dlswSdlcLsRemoteSap,
    dlswSdlcLsRowStatus
  }
STATUS current
DESCRIPTION
  "Conformance group for DLSw SDLC support."
 ::= { dlswGroups 16 }

-- .....
dlswNotificationGroup NOTIFICATION-GROUP

```

```
NOTIFICATIONS {
    dlswTrapTConnPartnerReject,
    dlswTrapTConnProtViolation,
    dlswTrapTConnUp,
    dlswTrapTConnDown,
    dlswTrapCircuitUp,
    dlswTrapCircuitDown
}
STATUS current
DESCRIPTION
    "Conformance group for DLsw notifications."
 ::= { dlswGroups 17 }
```

END

#### 4.0 Acknowledgements

This memo has been produced by the AIW DLSw MIB RIGlet, which is also recognized as the IETF DLSw MIB Working Group.

#### 5.0 References

- [1] Bartky, A., "Data Link Switching: Switch-to-Switch Protocol; AIW DLSw RIG: DLSw Closed Pages, DLSw Standard Version 1", RFC 1795, Sync Research Inc., April 1995.
- [2] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1902, January 1996.
- [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] McCloghrie, K., and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets - MIB-II", STD 17, RFC 1213, Hughes LAN Systems, Performance Systems International, March 1991.
- [5] 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.
- [6] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, January 1996.
- [7] IEEE Project, "ANSI/IEEE P802.1D", 1993
- [8] McCloghrie, K., and F. Kastenholz, "Evolution of the Interfaces Group of MIB-II", RFC 1573, Hughes LAN Systems, FTP Software, January 1994.
- [9] Hilgeman, J., S. Nix, A. Bartky, and W. Clark, "Definitions of Managed Objects for SNA Data Link Control (SDLC) using SMiv2", RFC 1747, Apertus Technologies, Inc., Metaplex, Inc., Sync Research, Inc., cisco Systems, Inc., January 1995

## 6.0 Security Considerations

Security issues are not discussed in this memo.

## 7.0 Authors' Addresses

David D. Chen  
IBM Networking Systems  
P. O. Box 12195  
Research Triangle Park, NC 27709  
US

Phone: +1 919 254 6182  
EMail: dchen@vnet.ibm.com

Peter W. Gayek  
IBM Networking Systems  
P. O. Box 12195  
Research Triangle Park, NC 27709  
US

Phone: +1 919 254 1808  
EMail: gayek@vnet.ibm.com

Shannon Nix  
Metaplex, Inc.

7025 Kit Creek Road  
P. O. Box 14987  
Research Triangle Park, NC 27709  
US

Phone: +1 919 472 2388  
EMail: snix@metaplex.com

