              Connectionless Network Protocol (ISO 8473)
                                and
              End System to Intermediate System (ISO 9542)
                      Management Information Base

                          Table of Contents

1. Status of this Memo

   This memo defines an experimental portion of the Management
   Information Base (MIB) for use with network management protocols in
   TCP/IP-based internets.

   This memo does not specify a standard for the Internet community.
   However, after experimentation, if sufficient consensus is reached in
   the Internet community, then a subsequent revision of this document
   may be placed in the Internet-standard MIB.

   Distribution of this memo is unlimited.

2. Historical Perspective

    As reported in RFC 1052, IAB Recommendations for the Development of
    Internet Network Management Standards [1], a two-prong strategy for
    network management of TCP/IP-based internets was undertaken. In the
    short-term, the Simple Network Management Protocol (SNMP), defined in
    RFC 1067, was to be used to manage nodes in the Internet community.
    In the long-term, the use of the OSI network management framework was
    to be examined. Two documents were produced to define the management
    information: RFC 1065, which defined the Structure of Management
    Information (SMI), and RFC 1066, which defined the Management
    Information Base (MIB). Both of these documents were designed so as
    to be compatible with both the SNMP and the OSI network management
    framework.

    This strategy was quite successful in the short-term: Internet-based
    network management technology was fielded, by both the research and
    commercial communities, within a few months. As a result of this,
    portions of the Internet community became network manageable in a
    timely fashion.

    As reported in RFC 1109, Report of the Second Ad Hoc Network
    Management Review Group [2], the requirements of the SNMP and the OSI
    network management frameworks were more different than anticipated.
    As such, the requirement for compatibility between the SMI/MIB and
    both frameworks was suspended. This action permitted the operational
    network management framework, based on the SNMP, to respond to new
    operational needs in the Internet community by producing MIB-II.

    In May of 1990, the core documents were elevated to "Standard
    Protocols" with "Recommended" status. As such, the Internet-
    standard network management framework consists of: Structure and
    Identification of Management Information for TCP/IP-based internets,
    RFC 1155 [3], which describes how managed objects contained in the
    MIB are defined; Management Information Base for Network Management
    of TCP/IP-based internets, which describes the managed objects
    contained in the MIB, RFC 1156 [4]; and, the Simple Network
    Management Protocol, RFC 1157 [5], which defines the protocol used to
    manage these objects.

    Consistent with the IAB directive to produce simple, workable systems
    in the short-term, the list of managed objects defined in the
    Internet-standard MIB was derived by taking only those elements which
    are considered essential. However, the SMI defined three
    extensibility mechanisms: one, the addition of new standard objects
    through the definitions of new versions of the MIB; two, the addition
    of widely-available but non-standard objects through the experimental
    subtree; and three, the addition of private objects through the

Satz   [Page 2]

enterprises subtree. Such additional objects can not only be used
for vendor-specific elements, but also for experimentation as
required to further the knowledge of which other objects are
essential.

Since the publication of the Internet-standard MIB, experience has
lead to a new document, termed MIB-II [6], being defined.

This memo defines extensions to the MIB using the second method. It
contains definitions of managed objects used for experimentation.
After experimentation, if sufficient consensus is reached in the
Internet community, then a subsequent revision of this memo may be
placed in the Internet-standard MIB.

3. Objects

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

The syntax of an object type defines the abstract data structure
corresponding to that object type. The ASN.1 language is used for
this purpose. However, the SMI [3] purposely restricts the ASN.1
constructs which may be used. These restrictions are explicitly made
for simplicity.

The encoding of an object type is simply how that object type is
represented using the object type's syntax. Implicitly tied to the
notion of an object type's syntax and encoding is how the object type
is represented when being transmitted on the network.

This SMI specifies the use of the basic encoding rules of ASN.1 [8],
subject to the additional requirements imposed by the SNMP.

3.1. Format of Definitions

The next section contains the specification of all object types
contained in the MIB. Following the conventions of the companion
memo, the object types are defined using the following fields:

OBJECT:

     A textual name, termed the OBJECT DESCRIPTOR, for the
     object type, along with its corresponding OBJECT
     IDENTIFIER.

Syntax:
     The abstract syntax for the object type, presented using
     ASN.1. This must resolve to an instance of the ASN.1
     type ObjectSyntax defined in the SMI.

Definition:
     A textual description of the semantics of the object
     type. Implementations should ensure that their
     interpretation of the object type fulfills this
     definition since this MIB is intended for use in multi-
     vendor environments. As such it is vital that object
     types have consistent meaning across all machines.

Access:
     A keyword, one of read-only, read-write, write-only, or
     not-accessible. Note that this designation specifies the
     minimum level of support required. As a local matter,
     implementations may support other access types (e.g., an
     implementation may elect to permitting writing a variable
     marked herein as read-only). Further, protocol-specific
     "views" (e.g., those implied by an SNMP community) may
     make further restrictions on access to a variable.

Status:
     A keyword, one of mandatory, optional, obsolete, or
     deprecated. Use of deprecated implies mandatory status.

4. Object Definitions

          CLNS-MIB DEFINITIONS ::= BEGIN

          IMPORTS
                  experimental, OBJECT-TYPE, Counter
                      FROM RFC1155-SMI;

          -- new type of NetworkAddress

          ClnpAddress ::=
              [APPLICATION 5]
                  IMPLICIT OCTET STRING (SIZE (1..21))

```
        clns    OBJECT IDENTIFIER ::=   { experimental 1 }

        clnp    OBJECT IDENTIFIER ::=   { clns 1 }
        error   OBJECT IDENTIFIER ::=   { clns 2 }
        echo    OBJECT IDENTIFIER ::=   { clns 3 }
        es-is   OBJECT IDENTIFIER ::=   { clns 4 }

        END
```

   These objects can be used when the ISO Connectionless-mode Network
   Protocol [9] and End System to Intermediate System [10] protocols are
   present. No assumptions are made as to what underlying protocol is
   being used to carry the SNMP.

   This memo uses the string encoding of [11] to textually describe OSI
   addresses.

   The ASN.1 type ClnpAddress is used to denote an OSI address. This
   consists of a string of octets. The first octet of the string
   contains a binary value in the range of 0..20, and indicates the the
   length in octets of the NSAP. Following the first octet, is the
   NSAP, expressed in concrete binary notation, starting with the most
   significant octet. A zero- length NSAP is used as a "special"
   address meaning "the default NSAP" (analogous to the IP address of
   0.0.0.0). Such an NSAP is encoded as a single octet, containing the
   value 0.

   All other NSAPs are encoded in at least 2 octets.

4.1. The CLNP Group

   Implementation is experimental and is recommended for all systems
   that support a CLNP.

```
        OBJECT:
        -------
            clnpForwarding { clnp 1 }

        Syntax:
            INTEGER {
                is(1),   -- entity is an intermediate system
                es(2),   -- entity is an end system and does not
                            forward PDUs
            }

        Definition:
            The indication of whether this entity is active as an
```

          intermediate or end system. Only intermediate systems
          will forward PDUs onward that are not addressed to them.

     Access:
          read-write.

     Status:
          mandatory.

     OBJECT:
     ----
          clnpDefaultLifeTime { clnp 2 }

     Syntax:
          INTEGER

     Definition:
          The default value inserted into the Lifetime field of the
          CLNP PDU header of PDUs sourced by this entity.

     Access:
          read-write.

     Status:
          mandatory.

     OBJECT:
     ----
          clnpInReceives { clnp 3 }

     Syntax:
          Counter

     Definition:
          The total number of input PDUs received from all
          connected network interfaces running CLNP, including
          errors.

     Access:
          read-only.

     Status:
          mandatory.

        OBJECT:
        ----
            clnpInHdrErrors { clnp 4 }

        Syntax:
            Counter

        Definition:
            The number of input PDUs discarded due to errors in the
            CLNP header, including bad checksums, version mismatch,
            lifetime exceeded, errors discovered in processing
            options, etc.

        Access:
            read-only.

        Status:
            mandatory.

        OBJECT:
        ----
            clnpInAddrErrors { clnp 5 }

        Syntax:
            Counter

        Definition:
            The number of input PDUs discarded because the NSAP
            address in the CLNP header's destination field was not a
            valid NSAP to be received at this entity. This count
            includes addresses not understood. For end systems, this
            is a count of PDUs which arrived with a destination NSAP
            which was not local.

        Access:
            read-only.

        Status:
            mandatory.

        OBJECT:
        ----
            clnpForwPDUs { clnp 6 }

        Syntax:
            Counter

Definition:
     The number of input PDUs for which this entity was not
     the final destination and which an attempt was made to
     forward them onward.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInUnknownNLPs { clnp 7 }

Syntax:
     Counter

Definition:
     The number of locally-addressed PDUs successfully
     received but discarded because the network layer protocol
     was unknown or unsupported (e.g., not CLNP or ES-IS).

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInUnknownULPs { clnp 8 }

Syntax:
     Counter

Definition:
     The number of locally-addressed PDUs successfully
     received but discarded because the upper layer protocol
     was unknown or unsupported (e.g., not TP4).

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
    clnpInDiscards { clnp 9 }

Syntax:
    Counter

Definition:
    The number of input CLNP PDUs for which no problems were
    encountered to prevent their continued processing, but
    were discarded (e.g., for lack of buffer space). Note that
    this counter does not include any PDUs discarded while
    awaiting re-assembly.

Access:
    read-only.

Status:
    mandatory.

OBJECT:
----
    clnpInDelivers { clnp 10 }

Syntax:
    Counter

Definition:
    The total number of input PDUs successfully delivered to
    the CLNS transport user.

Access:
    read-only.

Status:
    mandatory.

OBJECT:
----
    clnpOutRequests { clnp 11 }

Syntax:
    Counter

Definition:
    The total number of CLNP PDUs which local CLNS user

          protocols supplied to CLNP for transmission requests.
          This counter does not include any PDUs counted in
          clnpForwPDUs.

     Access:
          read-only.

     Status:
          mandatory.

     OBJECT:
     ----
          clnpOutDiscards { clnp 12 }

     Syntax:
          Counter

     Definition:
          The number of output CLNP PDUs for which no other problem
          was encountered to prevent their transmission but were
          discarded (e.g., for lack of buffer space). Note this
          counter includes PDUs counted in clnpForwPDUs.

     Access:
          read-only.

     Status:
          mandatory.

     OBJECT:
     ----
          clnpOutNoRoutes { clnp 13 }

     Syntax:
          Counter

     Definition:
          The number of CLNP PDUs discarded because no route could
          be found to transmit them to their destination. This
          counter includes any PDUs counted in clnpForwPDUs.

     Access:
          read-only.

     Status:
          mandatory.

OBJECT:
----
     clnpReasmTimeout { clnp 14 }

Syntax:
     INTEGER

Definition:
     The maximum number of seconds which received segments are
     held while they are awaiting reassembly at this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpReasmReqds { clnp 15 }

Syntax:
     Counter

Definition:
     The number of CLNP segments received which needed to be
     reassembled at this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpReasmOKs { clnp 16 }

Syntax:
     Counter

Definition:
     The number of CLNP PDUs successfully re-assembled at this
     entity.

     Access:
          read-only.

     Status:
          mandatory.

     OBJECT:
     ----
          clnpReasmFails { clnp 17 }

     Syntax:
          Counter

     Definition:
          The number of failures detected by the CLNP reassembly
          algorithm (for any reason: timed out, buffer size, etc).

     Access:
          read-only.

     Status:
          mandatory.

     OBJECT:
     ----
          clnpSegOKs { clnp 18 }

     Syntax:
          Counter

     Definition:
          The number of CLNP PDUs that have been successfully
          segmented at this entity.

     Access:
          read-only.

     Status:
          mandatory.

     OBJECT:
     ----
          clnpSegFails { clnp 19 }

        Syntax:
             Counter

        Definition:
             The number of CLNP PDUs that have been discarded because
             they needed to be fragmented at this entity but could
             not.

        Access:
             read-only.

        Status:
             mandatory.

        OBJECT:
        ----
             clnpSegCreates { clnp 20 }

        Syntax:
             Counter

        Definition:
             The number of CLNP PDU segments that have been generated
             as a result of segmentation at this entity.

        Access:
             read-only.

        Status:
             mandatory.

        OBJECT:
        ----
             clnpInOpts { clnp 25 }

        Syntax:
             Counter

        Definition:
             The number of CLNP PDU segments that have been input with
             options at this entity.

        Access:
             read-only.

        Status:
             mandatory.

        OBJECT:
        ----
             clnpOutOpts { clnp 26 }

        Syntax:
             Counter

        Definition:
             The number of CLNP PDU segments that have been generated
             with options by this entity.

        Access:
             read-only.

        Status:
             mandatory.

4.1.1. The CLNP Interfaces table

   The CLNP Interfaces table contains information on the entity's
   interfaces which are running the CLNP.

        OBJECT:
        ----
             clnpAddrTable { clnp 21 }

        Syntax:
             SEQUENCE OF ClnpAddrEntry

        Definition:
             The table of addressing information relevant to this
             entity's CLNP addresses.

        Access:
             not-accessible.

        Status:
             mandatory.

        OBJECT:
        ----
             clnpAddrEntry { clnpAddrTable 1 }

```
Syntax:
     ClnpAddrEntry :: = SEQUENCE {
          clnpAdEntAddr
               ClnpAddress,
          clnpAdEntIfIndex
               INTEGER,
          clnpAdEntReasmMaxSize
               INTEGER (0..65535)
     }
```

Definition:
     The addressing information for one of this entity's CLNP
     addresses.

Access:
     not-accessible.

Status:
     mandatory.

OBJECT:
----
     clnpAdEntAddr { clnpAddrEntry 1 }

Syntax:
     ClnpAddress

Definition:
     The CLNP address to which this entry's addressing
     information pertains.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpAdEntIfIndex { clnpAddrEntry 2 }

Syntax:
     INTEGER

Definition:
     The index value which uniquely identifies the interface

```

                to which this entry is applicable. The interface
                identified by a particular value of this index is the
                same interface as identified by the same value of
                ifIndex.

          Access:
                read-only.

          Status:
                mandatory.


          OBJECT:
          -------
                clnpAdEntReasmMaxSize { clnpAddrEntry 3 }

          Syntax:
                INTEGER (0..65535)

          Definition:
                The size of the largest CLNP PDU which this entity can
                re-assemble from incoming CLNP segmented PDUs received on
                this interface.

          Access:
                read-only.

          Status:
                mandatory.

4.1.2.  The CLNP Routing table

     The CLNP Routing table contains an entry for each route known to the
     entity.


          OBJECT:
          -------
                clnpRoutingTable { clnp 22 }

          Syntax:
                SEQUENCE OF ClnpRouteEntry

          Definition:
                This entity's CLNP routing table.

          Access:
                not-accessible.

```
Status:
    mandatory.

OBJECT:
----
    clnpRouteEntry { clnpRoutingTable 1 }

Syntax:
    ClnpRouteEntry ::= SEQUENCE {
          clnpRouteDest
              ClnpAddress,
          clnpRouteIfIndex
              INTEGER,
          clnpRouteMetric1
              INTEGER,
          clnpRouteMetric2
              INTEGER,
          clnpRouteMetric3
              INTEGER,
          clnpRouteMetric4
              INTEGER,
          clnpRouteNextHop
              ClnpAddress,
          clnpRouteType
              INTEGER,
          clnpRouteProto
              INTEGER,
          clnpRouteAge
              INTEGER
    }

Definition:
    A route to a particular destination.

Access:
    not-accessible.

Status:
    mandatory.

OBJECT:
----
    clnpRouteDest { clnpRouteEntry 1 }

Syntax:
    ClnpAddress
```

Definition:
     The destination CLNP address of this route.

Access:
     read-write.

Status:
     mandatory.


OBJECT:
-------
     clnpRouteIfIndex { clnpRouteEntry 2 }

Syntax:
     INTEGER

Definition:
     The index value which uniquely identifies the local
     interface through which the next hop of this route should
     be reached. The interface identified by a particular
     value of this index is the same as identified by the same
     value of ifIndex.

Access:
     read-write.

Status:
     mandatory.


OBJECT:
-------
     clnpRouteMetric1 { clnpRouteEntry 3 }

Syntax:
     INTEGER

Definition:
     The primary routing metric for this route. The semantics
     of this metric are determined by the routing-protocol
     specified in the route's clnpRouteProto value. If this
     metric is not used, its value should be set to -1.

Access:
     read-write.

          Status:
               mandatory.


          OBJECT:
          -------
               clnpRouteMetric2 { clnpRouteEntry 4 }

          Syntax:
               INTEGER

          Definition:
               An alternate routing metric for this route.  The
               semantics of this metric are determined by the routing-
               protocol specified in the route's clnpRouteProto value.
               If this metric is not used, its value should be set to
               -1.

          Access:
               read-write.

          Status:
               mandatory.


          OBJECT:
          -------
               clnpRouteMetric3 { clnpRouteEntry 5 }

          Syntax:
               INTEGER

          Definition:
               An alternate routing metric for this route.  The
               semantics of this metric are determined by the routing-
               protocol specified in the route's clnpRouteProto value.
               If this metric is not used, its value should be set to
               -1.

          Access:
               read-write.

          Status:
               mandatory.

```
     OBJECT:
     -------
         clnpRouteMetric4 { clnpRouteEntry 6 }

     Syntax:
         INTEGER

     Definition:
         An alternate routing metric for this route.  The
         semantics of this metric are determined by the routing-
         protocol specified in the route's clnpRouteProto value.
         If this metric is not used, its value should be set to
         -1.

     Access:
         read-write.

     Status:
         mandatory.


     OBJECT:
     -------
         clnpRouteNextHop { clnpRouteEntry 7 }

     Syntax:
         ClnpAddress

     Definition:
         The CLNP address of the next hop of this route.

     Access:
         read-write.

     Status:
         mandatory.


     OBJECT:
     -------
         clnpRouteType { clnpRouteEntry 8 }

     Syntax:
         INTEGER {
               other(1),      -- none of the following

               invalid(2),    -- an invalidated route
```

```
                              -- route to directly
                 direct(3),   -- connected (sub-)network

                              -- route to a non-local
                 remote(4)    -- host/network/sub-network
              }
```

Definition:
     The type of route.

Setting this object to the value invalid(2) has the effect of
invaliding the corresponding entry in the clnpRoutingTable.
That is, it effectively dissasociates the destination
identified with said entry from the route identified with said
entry. It is an implementation-specific matter as to whether
the agent removes an invalidated entry from the table.
Accordingly, management stations must be prepared to receive
tabular information from agents that corresponds to entries
not currently in use. Proper interpretation of such entries
requires examination of the relevant clnpRouteType object.

Access:
     read-write.

Status:
     mandatory.

OBJECT:
----
     clnpRouteProto { clnpRouteEntry 9 }

Syntax:
     INTEGER {
          other(1),   -- none of the following

                              -- non-protocol information
                              --   e.g., manually
          local(2),   --   configured entries

                              -- set via a network
          netmgmt(3), --   management protocol

                              -- similar to ipRouteProto
                              -- but omits several IP-specific
                              -- protocols

          is-is(9),
```

```
                    ciscoIgrp(11),
                    bbnSpfIgp(12),
                    ospf(13),
                    bgp(14)
              }
```

          Definition:
              The routing mechanism via which this route was learned.
              Inclusion of values for gateway routing protocols is not
              intended to imply that hosts should support those
              protocols.

          Access:
              read-write.

          Status:
              mandatory.

          OBJECT:
          ----
              clnpRouteAge { clnpRouteEntry 10 }

          Syntax:
              INTEGER

          Definition:
              The number of seconds since this route was last updated
              or otherwise determined to be correct.   Note that no
              semantics of "too old" can be implied except through
              knowledge of the routing protocol by which the route was
              learned.

          Access:
              read-write.

          Status:
              mandatory.

4.1.3. The CLNP Address Translation Tables

   The Address Translation tables contain the CLNP address to physical
   address equivalences. Some interfaces do not use translation tables
   for determining address equivalences; if all interfaces are of this
   type, then the Address Translation table is empty, i.e., has zero
   entries.

         OBJECT:
         ----
             clnpNetToMediaTable { clnp 23 }

         Syntax:
             SEQUENCE OF ClnpNetToMediaEntry

         Definition:
             The CLNP Address Translation table used for mapping from
             CLNP addresses to physical addresses.

         Access:
              not-accessible

         Status:
             mandatory.

         OBJECT:
         ----
             clnpNetToMediaEntry { clnpNetToMediaTable 1 }

         Syntax:
             ClnpNetToMediaEntry ::= SEQUENCE {
                 clnpNetToMediaIfIndex
                     INTEGER,
                 clnpNetToMediaPhysAddress
                     OCTET STRING,
                 clnpNetToMediaNetAddress
                     ClnpAddress,
                 clnpNetToMediaType
                     INTEGER,
                 clnpNetToMediaAge
                     INTEGER,
                 clnpNetToMediaHoldTime
                     INTEGER
             }

         Definition:
             Each entry contains one CLNP address to "physical"
             address equivalence.

         Access:
             not-accessible.

         Status:
             mandatory.

        OBJECT:
        ----
            clnpNetToMediaIfIndex { clnpNetToMediaEntry 1 }

        Syntax:
            INTEGER

        Definition:
            The interface on which this entry's equivalence is
            effective. The interface identified by a particular
            value of this index is the same interface as identified
            by the same value of ifIndex.

        Access:
            read-write.

        Status:
            mandatory.

        OBJECT:
        ----
            clnpNetToMediaPhysAddress { clnpNetToMediaEntry 2 }

        Syntax:
            OCTET STRING

        Definition:
            The media-dependent "physical" address.

        Access:
            read-write.

        Status:
            mandatory.

        OBJECT:
        ----
            clnpNetToMediaNetAddress { clnpNetToMediaEntry 3 }

        Syntax:
            ClnpAddress

        Definition:
            The CLNP address corresponding to the media-dependent
            "physical" address.

Access:
    read-write.

Status:
    mandatory.

OBJECT:
----
    clnpNetToMediaType { clnpNetToMediaEntry 4 }

Syntax:
    INTEGER {
        other(1),   -- none of the following
        invalid(2),   -- an invalidated mapping
        dynamic(3),
        static(4)
    }

Definition:
    The type of mapping.

Setting this object to the value invalid(2) has the effect of
invalidating the corresponding entry in the
clnpNetToMediaTable. That is, it effectively dissassociates
the interface identified with said entry from the mapping
identified with said entry. It is an implementation-specific
matter as to whether the agent removes an invalidated entry
from the table. Accordingly, management stations must be
prepared to receive tabular information from agents that
corresponds to entries not currently in use. Proper
interpretation of such entries requires examination of the
relevant clnpNetToMediaType object.

Access:
    read-write.

Status:
    mandatory.

OBJECT:
----
    clnpNetToMediaAge { clnpNetToMediaEntry 5 }

Syntax:
    INTEGER

        Definition:
             The number of seconds since this entry was last updated
             or otherwise determined to be correct. Note that no
             semantics of "too old" can be implied except through
             knowledge of the type of entry.

        Access:
             read-write.

        Status:
             mandatory.

        OBJECT:
        ----
             clnpNetToMediaHoldTime { clnpNetToMediaEntry 6 }

        Syntax:
             INTEGER

        Definition:
             The time in seconds this entry will be valid. Static
             entries should always report this field as -1.

        Access:
             read-write.

        Status:
             mandatory.

        OBJECT:
        ----
             clnpMediaToNetTable { clnp 24 }

        Syntax:
             SEQUENCE OF ClnpMediaToNetEntry

        Definition:
             The CLNP Address Translation table used for mapping from
             physical addresses to CLNP addresses.

        Access:
             not-accessible.

        Status:
             mandatory.

```
            OBJECT:
            -------
                 clnpMediaToNetEntry { clnpMediaToNetTable 1 }

            Syntax:
                 ClnpMediaToNetEntry ::= SEQUENCE {
                      clnpMediaToNetIfIndex
                           INTEGER,
                      clnpMediaToNetNetAddress
                           ClnpAddress,
                      clnpMediaToNetPhysAddress
                           OCTET STRING,
                      clnpMediaToNetType
                           INTEGER,
                      clnpMediaToNetAge
                           INTEGER,
                      clnpMediaToNetHoldTime
                           INTEGER
                 }

            Definition:
                 Each entry contains on ClnpAddress to "physical" address
                 equivalence.

            Access:
                 not-accessible.

            Status:
                 mandatory.


            OBJECT:
            -------
                 clnpMediaToNetIfIndex { clnpMediaToNetEntry 1 }

            Syntax:
                 INTEGER

            Definition:
                 The interface on which this entry's equivalence is
                 effective.  The interface identified by a particular
                 value of this index is the same interface as identified
                 by the same value of ifIndex.

            Access:
                 read-write.
```

Status:
     mandatory.

OBJECT:
----
     clnpMediaToNetAddress { clnpMediaToNetEntry 2 }

Syntax:
     ClnpAddress

Definition:
     The ClnpAddress corresponding to the media-dependent
     "physical" address.

Access:
     read-write.

Status:
     mandatory.

OBJECT:
----
     clnpMediaToNetPhysAddress { clnpMediaToNetEntry 3 }

Syntax:
     OCTET STRING

Definition:
     The media-dependent "physical" address.

Access:
     read-write.

Status:
     mandatory.

OBJECT:
----
     clnpMediaToNetType { clnpMediaToNetEntry 4 }

Syntax:
     INTEGER {
          other(1),   -- none of the following
          invalid(2),   -- an invalidated mapping
          dynamic(3),

                    static(4)
               }

          Definition:
               The type of mapping.

          Setting this object to the value invalid(2) has the effect of
          invalidating the corresponding entry in the
          clnpMediaToNetTable. That is, it effectively dissassociates
          the interface identified with said entry from the mapping
          identified with said entry. It is an implementation-specific
          matter as to whether the agent removes an invalidated entry
          from the table. Accordingly, management stations must be
          prepared to receive tabular information from agents that
          corresponds to entries not currently in use. Proper
          interpretation of such entries requires examination of the
          relevant clnpMediaToNetType object.

          Access:
               read-write.

          Status:
               mandatory.

          OBJECT:
          ----
               clnpMediaToNetAge { clnpMediaToNetEntry 5 }

          Syntax:
               INTEGER

          Definition:
               The number of seconds since this entry was last updated
               or otherwise determined to be correct. Note that no
               semantics of "too old" can be implied except through
               knowledge of the type of entry.

          Access:
               read-write.

          Status:
               mandatory.

          OBJECT:
          ----
               clnpMediaToNetHoldTime { clnpMediaToNetEntry 6 }

          Syntax:
               INTEGER

          Definition:
               The time in seconds this entry will be valid. Static
               entries should always report this field as -1.

          Access:
               read-write.

          Status:
               mandatory.

4.2. The CLNP Error Group

   This group records the CLNP Error protocol and is recommended for all
   systems which support CLNP.

          OBJECT:
          ----
               clnpInErrors { error 1 }

          Syntax:
               Counter

          Definition:
               The number of CLNP Error PDUs received by this entity.

          Access:
               read-only.

          Status:
               mandatory.

          OBJECT:
          ----
               clnpOutErrors { error 2 }

          Syntax:
               Counter

          Definition:
               The number of CLNP Error PDUs sent by this entity.

          Access:
               read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrUnspecs { error 3 }

Syntax:
     Counter

Definition:
     The number of unspecified CLNP Error PDUs received by
     this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrProcs { error 4 }

Syntax:
     Counter

Definition:
     The number of protocol procedure CLNP Error PDUs received
     by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrCksums { error 5 }

Syntax:
     Counter

Definition:
     The number of checksum CLNP Error PDUs received by this
     entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrCongests { error 6 }

Syntax:
     Counter

Definition:
     The number of congestion drop CLNP Error PDUs received by
     this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrHdrs { error 7 }

Syntax:
     Counter

Definition:
     The number of header syntax CLNP Error PDUs received by
     this entity.

Access:
     read-only.

Status:
     mandatory.

        OBJECT:
        ----
            clnpInErrSegs { error 8 }

        Syntax:
            Counter

        Definition:
            The number of segmentation disallowed CLNP Error PDUs
            received by this entity.

        Access:
            read-only.

        Status:
            mandatory.

        OBJECT:
        ----
            clnpInErrIncomps { error 9 }

        Syntax:
            Counter

        Definition:
            The number of incomplete PDU CLNP Error PDUs received by
            this entity.

        Access:
            read-only.

        Status:
            mandatory.

        OBJECT:
        ----
            clnpInErrDups { error 10 }

        Syntax:
            Counter

        Definition:
            The number of duplicate option CLNP Error PDUs received
            by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrUnreachDsts { error 11 }

Syntax:
     Counter

Definition:
     The number of unreachable destination CLNP Error PDUs
     received by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrUnknownDsts { error 12 }

Syntax:
     Counter

Definition:
     The number of unknown destination CLNP Error PDUs
     received by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrSRUnspecs { error 13 }

Syntax:
     Counter

Definition:
     The number of unspecified source route CLNP Error PDUs
     received by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrSRSyntaxes { error 14 }

Syntax:
     Counter

Definition:
     The number of source route syntax CLNP Error PDUs
     received by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrSRUnkAddrs { error 15 }

Syntax:
     Counter

Definition:
     The number of source route unknown address CLNP Error
     PDUs received by this entity.

Access:
     read-only.

Status:
     mandatory.

```
          OBJECT:
          -------
               clnpInErrSRBadPaths { error 16 }

          Syntax:
               Counter

          Definition:
               The number of source route bad path CLNP Error PDUs
               received by this entity.

          Access:
               read-only.

          Status:
               mandatory.


          OBJECT:
          -------
               clnpInErrHops { error 17 }

          Syntax:
               Counter

          Definition:
               The number of hop count exceeded CLNP Error PDUs received
               by this entity.

          Access:
               read-only.

          Status:
               mandatory.


          OBJECT:
          -------
               clnpInErrHopReassms { error 18 }

          Syntax:
               Counter

          Definition:
               The number of hop count exceeded while reassembling CLNP
               Error PDUs received by this entity.
```

          Access:
               read-only.

          Status:
               mandatory.

          OBJECT:
          ----
               clnpInErrUnsOptions { error 19 }

          Syntax:
               Counter

          Definition:
               The number of unsupported option CLNP Error PDUs received
               by this entity.

          Access:
               read-only.

          Status:
               mandatory.

          OBJECT:
          ----
               clnpInErrUnsVersions { error 20 }

          Syntax:
               Counter

          Definition:
               The number of version mismatch CLNP Error PDUs received
               by this entity.

          Access:
               read-only.

          Status:
               mandatory.

          OBJECT:
          ----
               clnpInErrUnsSecurities { error 21 }

Syntax:
     Counter

Definition:
     The number of unsupported security option CLNP Error PDUs
     received by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrUnsSRs { error 22 }

Syntax:
     Counter

Definition:
     The number of unsupported source route option CLNP Error
     PDUs received by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpInErrUnsRRs { error 23 }

Syntax:
     Counter

Definition:
     The number of unsupported record route option CLNP Error
     PDUs received by this entity.

Access:
     read-only.

Status:
     mandatory.

          OBJECT:
          ----
               clnpInErrInterferences { error 24 }

          Syntax:
               Counter

          Definition:
               The number of reassembly interference CLNP Error PDUs
               received by this entity.

          Access:
               read-only.

          Status:
               mandatory.

          OBJECT:
          ----
               clnpOutErrUnspecs { error 25 }

          Syntax:
               Counter

          Definition:
               The number of unspecified CLNP Error PDUs sent by this
               entity.

          Access:
               read-only.

          Status:
               mandatory.

          OBJECT:
          ----
               clnpOutErrProcs { error 26 }

          Syntax:
               Counter

          Definition:
               The number of protocol procedure CLNP Error PDUs sent by
               this entity.

    Access:
        read-only.

    Status:
        mandatory.

    OBJECT:
    ----
        clnpOutErrCksums { error 27 }

    Syntax:
        Counter

    Definition:
        The number of checksum CLNP Error PDUs sent by this
        entity.

    Access:
        read-only.

    Status:
        mandatory.

    OBJECT:
    ----
        clnpOutErrCongests { error 28 }

    Syntax:
        Counter

    Definition:
        The number of congestion drop CLNP Error PDUs sent by
        this entity.

    Access:
        read-only.

    Status:
        mandatory.

    OBJECT:
    ----
        clnpOutErrHdrs { error 29 }

Syntax:
     Counter

Definition:
     The number of header syntax CLNP Error PDUs sent by this
     entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpOutErrSegs { error 30 }

Syntax:
     Counter

Definition:
     The number of segmentation disallowed CLNP Error PDUs
     sent by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpOutErrIncomps { error 31 }

Syntax:
     Counter

Definition:
     The number of incomplete PDU CLNP Error PDUs sent by this
     entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
    clnpOutErrDups { error 32 }

Syntax:
    Counter

Definition:
    The number of duplicate option CLNP Error PDUs sent by
    this entity.

Access:
    read-only.

Status:
    mandatory.

OBJECT:
----
    clnpOutErrUnreachDsts { error 33 }

Syntax:
    Counter

Definition:
    The number of unreachable destination CLNP Error PDUs
    sent by this entity.

Access:
    read-only.

Status:
    mandatory.

OBJECT:
----
    clnpOutErrUnknownDsts { error 34 }

Syntax:
    Counter

Definition:
    The number of unknown destination CLNP Error PDUs sent by
    this entity.

Access:
      read-only.

Status:
      mandatory.

OBJECT:
----
      clnpOutErrSRUnspecs { error 35 }

Syntax:
      Counter

Definition:
      The number of unspecified source route CLNP Error PDUs
      sent by this entity.

Access:
      read-only.

Status:
      mandatory.

OBJECT:
----
      clnpOutErrSRSyntaxes { error 36 }

Syntax:
      Counter

Definition:
      The number of source route syntax CLNP Error PDUs sent by
      this entity.

Access:
      read-only.

Status:
      mandatory.

OBJECT:
----
      clnpOutErrSRUnkAddrs { error 37 }

Syntax:
     Counter

Definition:
     The number of source route unknown address CLNP Error
     PDUs sent by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpOutErrSRBadPaths { error 38 }

Syntax:
     Counter

Definition:
     The number of source route bad path CLNP Error PDUs sent
     by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpOutErrHopss { error 39 }

Syntax:
     Counter

Definition:
     The number of hop count exceeded CLNP Error PDUs sent by
     this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpOutErrHopReassms { error 40 }

Syntax:
     Counter

Definition:
     The number of hop count exceeded while reassembling CLNP
     Error PDUs sent by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpOutErrUnsOptions { error 41 }

Syntax:
     Counter

Definition:
     The number of unsupported option CLNP Error PDUs sent by
     this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpOutErrUnsVersions { error 42 }

Syntax:
     Counter

Definition:
     The number of version mismatch CLNP Error PDUs sent by
     this entity.

     Access:
          read-only.

     Status:
          mandatory.

     OBJECT:
     ----
          clnpOutErrUnsSecurities { error 43 }

     Syntax:
          Counter

     Definition:
          The number of unsupported security option CLNP Error PDUs
          sent by this entity.

     Access:
          read-only.

     Status:
          mandatory.

     OBJECT:
     ----
          clnpOutErrUnsSRs { error 44 }

     Syntax:
          Counter

     Definition:
          The number of unsupported source route option CLNP Error
          PDUs sent by this entity.

     Access:
          read-only.

     Status:
          mandatory.

     OBJECT:
     ----
          clnpOutErrUnsRRs { error 45 }

Syntax:
     Counter

Definition:
     The number of unsupported record route option CLNP Error
     PDUs sent by this entity.

Access:
     read-only.

Status:
     mandatory.

OBJECT:
----
     clnpOutErrInterferences { error 46 }

Syntax:
     Counter

Definition:
     The number of reassembly interference CLNP Error PDUs
     sent by this entity.

Access:
     read-only.

Status:
     mandatory.

4.3. The ESIS Group

   The ESIS group contains information about the End System Intermediate
   System protocol used to maintain neighbor reachibility information.
   Both ESs and ISs are expected to implement this group if they running
   a CLNP.

     OBJECT:
     ----
          esisESHin { es-is 1 }

     Syntax:
          Counter

     Definition:
          The number of ESH PDUs received by this entity.

        Access:
            read-only.

        Status:
            mandatory.

        OBJECT:
        ----
            esisESHout { es-is 2 }

        Syntax:
            Counter

        Definition:
            The number of ESH PDUs sent by this entity.

        Access:
            read-only.

        Status:
            mandatory.

        OBJECT:
        ----
            esisISHin { es-is 3 }

        Syntax:
            Counter

        Definition:
            The number of ISH PDUs received by this entity.

        Access:
            read-only.

        Status:
            mandatory.

        OBJECT:
        ----
            esisISHout { es-is 4 }

        Syntax:
            Counter

          Definition:
               The number of ISH PDUs sent by this entity.

          Access:
               read-only.

          Status:
               mandatory.

          OBJECT:
          ----
               esisRDUin { es-is 5 }

          Syntax:
               Counter

          Definition:
               The number of RDU PDUs received by this entity.

          Access:
               read-only.

          Status:
               mandatory.

          OBJECT:
          ----
               esisRDUout { es-is 6 }

          Syntax:
               Counter

          Definition:
               The number of RDU PDUs sent by this entity.

          Access:
               read-only.

          Status:
               mandatory.

5. Definitions

```
         CLNS-MIB DEFINITIONS ::= BEGIN

         IMPORTS
                 experimental, OBJECT-TYPE, Counter
                         FROM RFC1155-SMI;

         -- new type of NetworkAddress

         ClnpAddress ::=
             [APPLICATION 5]
                 IMPLICIT OCTET STRING (SIZE (1..21))


         clns    OBJECT IDENTIFIER ::=   { experimental 1 }

         clnp    OBJECT IDENTIFIER ::=   { clns 1 }
         error   OBJECT IDENTIFIER ::=   { clns 2 }
         echo    OBJECT IDENTIFIER ::=   { clns 3 }
         es-is   OBJECT IDENTIFIER ::=   { clns 4 }


         -- the General CLNP group

         clnpForwarding OBJECT-TYPE
                 SYNTAX   INTEGER {
                             is(1),  -- entity is an
                                     -- intermediate system
                             es(2)   -- entity is an end system
                                     -- and does not forward pdus
                         }
                 ACCESS   read-write
                 STATUS   mandatory
                 ::= { clnp 1 }

         clnpDefaultLifeTime OBJECT-TYPE
                 SYNTAX   INTEGER
                 ACCESS   read-write
                 STATUS   mandatory
                 ::= { clnp 2 }

         clnpInReceives OBJECT-TYPE
                 SYNTAX  Counter
                 ACCESS   read-only
                 STATUS   mandatory
                 ::= { clnp 3 }
```

```
        clnpInHdrErrors OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 4 }

        clnpInAddrErrors OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 5 }

        clnpForwPDUs OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 6 }

        clnpInUnknownNLPs OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 7 }

        clnpInUnknownULPs OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 8 }

        clnpInDiscards OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 9 }

        clnpInDelivers OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 10 }

        clnpOutRequests OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 11 }
```

```
        clnpOutDiscards OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 12 }

        clnpOutNoRoutes OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 13 }

        clnpReasmTimeout OBJECT-TYPE
                SYNTAX  INTEGER
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 14 }

        clnpReasmReqds  OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 15 }

        clnpReasmOKs    OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 16 }

        clnpReasmFails  OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 17 }

        clnpSegOKs      OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 18 }

        clnpSegFails    OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 19 }
```

```
        clnpSegCreates  OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 20 }

        clnpInOpts      OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 25 }

        clnpOutOpts     OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnp 26 }

        -- the CLNP Interface table

        clnpAddrTable   OBJECT-TYPE
                SYNTAX  SEQUENCE OF ClnpAddrEntry
                ACCESS  not-accessible
                STATUS  mandatory
                ::= { clnp 21 }

        clnpAddrEntry   OBJECT-TYPE
                SYNTAX  ClnpAddrEntry
                ACCESS  not-accessible
                STATUS  mandatory
        --      INDEX   { clnpAdEntAddr }
                ::= { clnpAddrTable 1 }

        ClnpAddrEntry ::= SEQUENCE {
            clnpAdEntAddr
                ClnpAddress,
            clnpAdEntIfIndex
                INTEGER,
            clnpAdEntReasmMaxSize
                INTEGER (0..65535)
        }

        clnpAdEntAddr OBJECT-TYPE
                SYNTAX  ClnpAddress
                ACCESS  read-only
                STATUS  mandatory
                ::=  { clnpAddrEntry 1 }
```

```
        clnpAdEntIfIndex OBJECT-TYPE
                SYNTAX  INTEGER
                ACCESS  read-only
                STATUS  mandatory
                ::=  { clnpAddrEntry 2 }

        clnpAdEntReasmMaxSize OBJECT-TYPE
                SYNTAX  INTEGER (0..65535)
                ACCESS  read-only
                STATUS  mandatory
                ::= { clnpAddrEntry 3 }

        -- the CLNP Routing table

        clnpRoutingTable OBJECT-TYPE
                SYNTAX  SEQUENCE OF ClnpRouteEntry
                ACCESS  not-accessible
                STATUS  mandatory
                ::= { clnp 22 }

        clnpRouteEntry OBJECT-TYPE
                SYNTAX  ClnpRouteEntry
                ACCESS  not-accessible
                STATUS  mandatory
        --      INDEX   { clnpRouteDest }
                ::= { clnpRoutingTable 1 }

        ClnpRouteEntry ::= SEQUENCE {
            clnpRouteDest
                ClnpAddress,
            clnpRouteIfIndex
                INTEGER,
            clnpRouteMetric1
                INTEGER,
            clnpRouteMetric2
                INTEGER,
            clnpRouteMetric3
                INTEGER,
            clnpRouteMetric4
                INTEGER,
            clnpRouteNextHop
                ClnpAddress,
            clnpRouteType
                INTEGER,
            clnpRouteProto
                INTEGER,
            clnpRouteAge
                INTEGER
```

```
        }

        clnpRouteDest OBJECT-TYPE
                SYNTAX  ClnpAddress
                ACCESS  read-write
                STATUS  mandatory
                ::= { clnpRouteEntry 1 }

        clnpRouteIfIndex  OBJECT-TYPE
                SYNTAX  INTEGER
                ACCESS  read-write
                STATUS  mandatory
                ::= { clnpRouteEntry 2 }

        clnpRouteMetric1 OBJECT-TYPE
                SYNTAX  INTEGER
                ACCESS  read-write
                STATUS  mandatory
                ::= { clnpRouteEntry 3 }

        clnpRouteMetric2 OBJECT-TYPE
                SYNTAX  INTEGER
                ACCESS  read-write
                STATUS  mandatory
                ::= { clnpRouteEntry 4 }

        clnpRouteMetric3 OBJECT-TYPE
                SYNTAX  INTEGER
                ACCESS  read-write
                STATUS  mandatory
                ::= { clnpRouteEntry 5 }

        clnpRouteMetric4 OBJECT-TYPE
                SYNTAX  INTEGER
                ACCESS  read-write
                STATUS  mandatory
                ::= { clnpRouteEntry 6 }

        clnpRouteNextHop OBJECT-TYPE
                SYNTAX  ClnpAddress
                ACCESS  read-write
                STATUS  mandatory
                ::= { clnpRouteEntry 7 }

        clnpRouteType OBJECT-TYPE
                SYNTAX   INTEGER {
                            other(1),      -- none of the
                                           -- following
```

```
                             invalid(2),     -- an invalidated
                                             -- route

                                             -- route to directly
                             direct(3),      -- connected
                                             -- (sub-)network

                                             -- route to a
                                             -- non-local
                             remote(4)       -- host/network
                                             -- /sub-network
                    }
            ACCESS   read-write
            STATUS   mandatory
            ::= { clnpRouteEntry 8 }

        clnpRouteProto OBJECT-TYPE
            SYNTAX   INTEGER {
                             other(1),       -- none of the
                                             -- following

                                             -- non-protocol
                                             -- information
                                             --   e.g., manually
                             local(2),       --   configured entries

                                             -- set via a network
                             netmgmt(3),     --   management
                                             --   protocol

                                             -- similar to
                                             -- ipRouteProto
                                             -- but omits several
                                             -- IP-specific
                                             -- protocols
                             is-is(9),
                             ciscoIgrp(11),
                             bbnSpfIgp(12),
                             ospf(13),
                             bgp(14)
                    }
            ACCESS   read-only
            STATUS   mandatory
            ::= { clnpRouteEntry 9 }

        clnpRouteAge OBJECT-TYPE
            SYNTAX   INTEGER
            ACCESS   read-write
```

```
                STATUS  mandatory
                ::= { clnpRouteEntry 10 }

        -- the CLNP Address Translation tables

        clnpNetToMediaTable OBJECT-TYPE
                SYNTAX   SEQUENCE OF ClnpNetToMediaEntry
                ACCESS   not-accessible
                STATUS   mandatory
                ::= { clnp 23 }

        clnpNetToMediaEntry OBJECT-TYPE
                SYNTAX   ClnpNetToMediaEntry
                ACCESS   not-accessible
                STATUS   mandatory
        --      INDEX   { clnpNetToMediaIfIndex,
        --                clnpNetToMediaNetAddress }
                ::= { clnpNetToMediaTable 1 }

        ClnpNetToMediaEntry ::= SEQUENCE {
            clnpNetToMediaIfIndex
                INTEGER,
            clnpNetToMediaPhysAddress
                OCTET STRING,
            clnpNetToMediaNetAddress
                ClnpAddress,
            clnpNetToMediaType
                INTEGER,
            clnpNetToMediaAge
                INTEGER,
            clnpNetToMediaHoldTime
                INTEGER
        }

        clnpNetToMediaIfIndex OBJECT-TYPE
                SYNTAX   INTEGER
                ACCESS   read-write
                STATUS   mandatory
                ::= { clnpNetToMediaEntry 1 }

        clnpNetToMediaPhysAddress OBJECT-TYPE
                SYNTAX   OCTET STRING
                ACCESS   read-write
                STATUS   mandatory
                ::= { clnpNetToMediaEntry 2 }

        clnpNetToMediaNetAddress OBJECT-TYPE
                SYNTAX   ClnpAddress
```

```
                    ACCESS   read-write
                    STATUS   mandatory
                    ::= { clnpNetToMediaEntry 3 }

            clnpNetToMediaType OBJECT-TYPE
                    SYNTAX   INTEGER {
                                other(1),       -- none of the
                                                -- following
                                invalid(2),     -- an invalidated
                                                -- mapping
                                dynamic(3),
                                static(4)
                        }
                    ACCESS   read-write
                    STATUS   mandatory
                    ::= { clnpNetToMediaEntry 4 }

            clnpNetToMediaAge OBJECT-TYPE
                    SYNTAX   INTEGER
                    ACCESS   read-write
                    STATUS   mandatory
                    ::= { clnpNetToMediaEntry 5 }

            clnpNetToMediaHoldTime OBJECT-TYPE
                    SYNTAX   INTEGER
                    ACCESS   read-write
                    STATUS   mandatory
                    ::= { clnpNetToMediaEntry 6 }

            clnpMediaToNetTable OBJECT-TYPE
                    SYNTAX   SEQUENCE OF ClnpMediaToNetEntry
                    ACCESS   not-accessible
                    STATUS   mandatory
                    ::= { clnp 24 }

            clnpMediaToNetEntry OBJECT-TYPE
                    SYNTAX   ClnpMediaToNetEntry
                    ACCESS   not-accessible
                    STATUS   mandatory
            --      INDEX   { clnpMediaToNetIfIndex,
            --                clnpMediaToNetPhysAddress }
                    ::= { clnpMediaToNetTable 1 }

            ClnpMediaToNetEntry ::= SEQUENCE {
                clnpMediaToNetIfIndex
                    INTEGER,
                clnpMediaToNetNetAddress
                    ClnpAddress,
```

```
        clnpMediaToNetPhysAddress
            OCTET STRING,
        clnpMediaToNetType
            INTEGER,
        clnpMediaToNetAge
            INTEGER,
        clnpMediaToNetHoldTime
            INTEGER
    }

clnpMediaToNetIfIndex OBJECT-TYPE
        SYNTAX   INTEGER
        ACCESS   read-write
        STATUS   mandatory
        ::= { clnpMediaToNetEntry 1 }

clnpMediaToNetNetAddress OBJECT-TYPE
        SYNTAX   ClnpAddress
        ACCESS   read-write
        STATUS   mandatory
        ::= { clnpMediaToNetEntry 2 }

clnpMediaToNetPhysAddress OBJECT-TYPE
        SYNTAX   OCTET STRING
        ACCESS   read-write
        STATUS   mandatory
        ::= { clnpMediaToNetEntry 3 }

clnpMediaToNetType OBJECT-TYPE
        SYNTAX   INTEGER {
                    other(1),       -- none of the
                                    -- following
                    invalid(2),     -- an invalidated
                                    -- mapping
                    dynamic(3),
                    static(4)
                }
        ACCESS   read-write
        STATUS   mandatory
        ::= { clnpMediaToNetEntry 4 }

clnpMediaToNetAge OBJECT-TYPE
        SYNTAX   INTEGER
        ACCESS   read-write
        STATUS   mandatory
        ::= { clnpMediaToNetEntry 5 }

clnpMediaToNetHoldTime OBJECT-TYPE
```

```
                 SYNTAX   INTEGER
                 ACCESS   read-write
                 STATUS   mandatory
                 ::= { clnpMediaToNetEntry 6 }

         -- the CLNP Error Group

         clnpInErrors    OBJECT-TYPE
                 SYNTAX   Counter
                 ACCESS   read-only
                 STATUS   mandatory
                 ::= { error 1 }

         clnpOutErrors   OBJECT-TYPE
                 SYNTAX   Counter
                 ACCESS   read-only
                 STATUS   mandatory
                 ::= { error 2 }

         clnpInErrUnspecs OBJECT-TYPE
                 SYNTAX   Counter
                 ACCESS   read-only
                 STATUS   mandatory
                 ::= { error 3 }

         clnpInErrProcs  OBJECT-TYPE
                 SYNTAX   Counter
                 ACCESS   read-only
                 STATUS   mandatory
                 ::= { error 4 }

         clnpInErrCksums OBJECT-TYPE
                 SYNTAX   Counter
                 ACCESS   read-only
                 STATUS   mandatory
                 ::= { error 5 }

         clnpInErrCongests OBJECT-TYPE
                 SYNTAX   Counter
                 ACCESS   read-only
                 STATUS   mandatory
                 ::= { error 6 }

         clnpInErrHdrs   OBJECT-TYPE
                 SYNTAX   Counter
                 ACCESS   read-only
                 STATUS   mandatory
                 ::= { error 7 }
```

```
        clnpInErrSegs   OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 8 }

        clnpInErrIncomps OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 9 }

        clnpInErrDups   OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 10 }

        clnpInErrUnreachDsts OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 11 }

        clnpInErrUnknownDsts OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 12 }

        clnpInErrSRUnspecs OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 13 }

        clnpInErrSRSyntaxes OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 14 }

        clnpInErrSRUnkAddrs OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 15 }
```

```
clnpInErrSRBadPaths OBJECT-TYPE
        SYNTAX   Counter
        ACCESS   read-only
        STATUS   mandatory
        ::= { error 16 }

clnpInErrHops   OBJECT-TYPE
        SYNTAX   Counter
        ACCESS   read-only
        STATUS   mandatory
        ::= { error 17 }

clnpInErrHopReassms OBJECT-TYPE
        SYNTAX   Counter
        ACCESS   read-only
        STATUS   mandatory
        ::= { error 18 }

clnpInErrUnsOptions OBJECT-TYPE
        SYNTAX   Counter
        ACCESS   read-only
        STATUS   mandatory
        ::= { error 19 }

clnpInErrUnsVersions OBJECT-TYPE
        SYNTAX   Counter
        ACCESS   read-only
        STATUS   mandatory
        ::= { error 20 }

clnpInErrUnsSecurities OBJECT-TYPE
        SYNTAX   Counter
        ACCESS   read-only
        STATUS   mandatory
        ::= { error 21 }

clnpInErrUnsSRs OBJECT-TYPE
        SYNTAX   Counter
        ACCESS   read-only
        STATUS   mandatory
        ::= { error 22 }

clnpInErrUnsRRs OBJECT-TYPE
        SYNTAX   Counter
        ACCESS   read-only
        STATUS   mandatory
        ::= { error 23 }
```

```
        clnpInErrInterferences OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 24 }

        clnpOutErrUnspecs OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 25 }

        clnpOutErrProcs OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 26 }

        clnpOutErrCksums OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 27 }

        clnpOutErrCongests OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 28 }

        clnpOutErrHdrs OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 29 }

        clnpOutErrSegs  OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 30 }

        clnpOutErrIncomps OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 31 }
```

```
        clnpOutErrDups  OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 32 }

        clnpOutErrUnreachDsts OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 33 }

        clnpOutErrUnknownDsts OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 34 }

        clnpOutErrSRUnspecs OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 35 }

        clnpOutErrSRSyntaxes OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 36 }

        clnpOutErrSRUnkAddrs OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 37 }

        clnpOutErrSRBadPaths OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 38 }

        clnpOutErrHops  OBJECT-TYPE
                SYNTAX  Counter
                ACCESS  read-only
                STATUS  mandatory
                ::= { error 39 }
```

```
        clnpOutErrHopReassms OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 40 }

        clnpOutErrUnsOptions OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 41 }

        clnpOutErrUnsVersions OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 42 }

        clnpOutErrUnsSecurities OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 43 }

        clnpOutErrUnsSRs OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 44 }

        clnpOutErrUnsRRs         OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 45 }

        clnpOutErrInterferences OBJECT-TYPE
                SYNTAX   Counter
                ACCESS   read-only
                STATUS   mandatory
                ::= { error 46 }


        -- the CLNP Echo Group

        -- the ES-IS Group

        esisESHins      OBJECT-TYPE
```

```
                        SYNTAX  Counter
                        ACCESS  read-only
                        STATUS  mandatory
                        ::= { es-is 1 }

                esisESHouts     OBJECT-TYPE
                        SYNTAX  Counter
                        ACCESS  read-only
                        STATUS  mandatory
                        ::= { es-is 2 }

                esisISHins      OBJECT-TYPE
                        SYNTAX  Counter
                        ACCESS  read-only
                        STATUS  mandatory
                        ::= { es-is 3 }

                esisISHouts     OBJECT-TYPE
                        SYNTAX  Counter
                        ACCESS  read-only
                        STATUS  mandatory
                        ::= { es-is 4 }

                esisRDUins      OBJECT-TYPE
                        SYNTAX  Counter
                        ACCESS  read-only
                        STATUS  mandatory
                        ::= { es-is 5 }

                esisRDUouts     OBJECT-TYPE
                        SYNTAX  Counter
                        ACCESS  read-only
                        STATUS  mandatory
                        ::= { es-is 6 }

                END
```

6. Identification of OBJECT instances for use with the SNMP

   The names for all object types in the MIB are defined explicitly
   either in the Internet-standard MIB or in other documents which
   conform to the naming conventions of the SMI. The SMI requires that
   conformant management protocols define mechanisms for identifying
   individual instances of those object types for a particular network
   element.

   Each instance of any object type defined in the MIB is identified in
   SNMP operations by a unique name called its "variable name." In

general, the name of an SNMP variable is an OBJECT IDENTIFIER of the
form x.y, where x is the name of a non-aggregate object type defined
in the MIB and y is an OBJECT IDENTIFIER fragment that, in a way
specific to the named object type, identifies the desired instance.

This naming strategy admits the fullest exploitation of the semantics
of the powerful SNMP get-next operator, because it assigns names for
related variables so as to be contiguous in the lexicographical
ordering of all variable names known in the MIB.

The type-specific naming of object instances is defined below for a
number of classes of object types. Instances of an object type to
which none of the following naming conventions are applicable are
named by OBJECT IDENTIFIERs of the form x.0, where x is the name of
said object type in the MIB definition.

For example, suppose one wanted to identify an instance of the
variable sysDescr in the Internet-standard MIB. The object class for
sysDescr is:

     iso org dod internet mgmt mib system sysDescr
      1   3   6   1   2   1   1   1

Hence, the object type, x, would be 1.3.6.1.2.1.1.1 to which is
appended an instance sub-identifier of 0. That is, 1.3.6.1.2.1.1.1.0
identifies the one and only instance of sysDescr.

6.1. clnpAddrTable Object Type Names

The name of an CLNP-addressable network element, x, is the OBJECT
IDENTIFIER of the form z such that z is the value (in which each
octet of the ClnpAddress type is expressed as a sub-identifier of the
OBJECT IDENTIFIER) of that instance of the clnpAdEntAddr object type
associated with x.

For each object type, t, for which the defined name, n, has a prefix
of clnpAddrEntry, an instance, i, of t is named by an OBJECT
IDENTIFIER of the form n.y, where y is the name of the CLNP-
addressable network element about which i represents information.

For example, suppose one wanted to find the maximum reassembly size
of an entry in the CLNP interface table associated with an CLNP
address of NS+0504030201. Accordingly, clnpAdEntNetMask.5.5.4.3.2.1
would identify the desired instance.

6.2. clnpRoutingTable Object Type Names

The name of an CLNP route, x, is the OBJECT IDENTIFIER of the form z

such that z is the value (in which each octet of the ClnpAddress type
is expressed as a sub-identifier of the OBJECT IDENTIFIER) of that
instance of the clnpRouteDest object type associated with x.

For each object type, t, for which the defined name, n, has a prefix
of clnpRoutingEntry, an instance, i, of t is named by an OBJECT
IDENTIFIER of the form n.y, where y is the name of the CLNP route
about which i represents information.

For example, suppose one wanted to find the next hop of an entry in
the CLNP routing table associated with the destination of
NS+0504030201. Accordingly, clnpRouteNextHop.5.5.4.3.2.1 would
identify the desired instance.

At the option of the agent, multiple routes to the same destination
may be visible. To realize this, the agent, while required to return
a single entry for an CLNP route, x, of the form n.y, may also return
information about other routes to the same destination using the form
n.y.v, where v is a implementation-dependent small, non-negative
integer.

6.3.   clnpNetToMediaTable Object Type Names

The name of a cached CLNP address, x, is an OBJECT IDENTIFIER of the
form s.z, such that s is the value of that instance of the
clnpNetToMediaIfIndex object type associated with the entry and z is
the value of the CLNP address of the clnpNetToMediaNetAddress object
type associated with x, in which each octet of the ClnpAddress type
is expressed as a sub-identifier of the OBJECT IDENTIFIER.

For each object type, t, for which the defined name, n, has a prefix
of clnpNetToMediaEntry, an instance, i, of t is named by an OBJECT
IDENTIFIER of the form n.y, where y is the name of the cached CLNP
address about which i represents information.

For example, suppose one wanted to find the media address of an entry
in the address translation table associated with a CLNP address of
NS+0504030201 and interface 3. Accordingly,
clnpNetToMediaPhysAddress.3.5.5.4.3.2.1 would identify the desired
instance.

6.4.   clnpMediaToNetTable Object Type Names

The name of a cached media address, x, is an OBJECT IDENTIFIER of the
form s.z, such that s is the value of that instance of the
clnpMediaToNetIfIndex object type associated with the entry and z is
the value of the media address of the clnpMediaToNetMediaAddress
object type associated with x, in which each octet of the media

Satz   [Page 68]

   address is expressed as a sub- identifier of the OBJECT IDENTIFIER.

   For each object type, t, for which the defined name, n, has a prefix
   of clnpMediaToNetEntry, an instance, i, of t is named by an OBJECT
   IDENTIFIER of the form n.y, where y is the name of the cached media
   address about which i represents information.

   For example, suppose one wanted to find the CLNP address of an entry
   in the address translation table associated with a media address of
   08:00:20:00:38:ba and interface 3. Accordingly,
   clnpMediaToNetNetAddress.3.8.0.32.0.56.186 would identify the desired
   instance.

7. References

   [1] Cerf, V., "IAB Recommendations for the Development of Internet
        Network Management Standards", RFC 1052, IAB, April 1988.

   [2] Cerf, V., "Report of the Second Ad Hoc Network Management Review
        Group", RFC 1109, NRI, August 1989.

   [3] Rose, M., and K. McCloghrie, "Structure and Identification of
        Management Information for TCP/IP-based Internets", RFC 1155,
        Performance Systems International and Hughes LAN Systems, May
        1990.

   [4] McCloghrie, K., and M. Rose, "Management Information Base for
        Network Management of TCP/IP-based Internets", RFC 1156, Hughes
        LAN Systems and Performance Systems International, May 1990.

   [5] Case, J., M. Fedor, M. Schoffstall, and J. Davin, The Simple
        Network Management Protocol", RFC 1157, University of Tennessee
        at Knoxville, Performance Systems International, Performance
        Systems International, and the MIT Laboratory for Computer
        Science, May 1990.

   [6] Rose, M., "Management Information Base for Network Management of
        TCP/IP-based internets: MIB-II", RFC 1158, Performance Systems
        International, May 1990.

   [7] Information processing systems - Open Systems Interconnection,
        "Specification of Abstract Syntax Notation One (ASN.1)",
        International Organization for Standardization, International
        Standard 8824, December 1987.

   [8] Information processing systems - Open Systems Interconnection,
        "Specification of Basic Encoding Rules for Abstract Notation One
        (ASN.1)", International Organization for Standardization,

        International Standard 8825, December 1987.

   [9]   Information processing systems - Data Communications - "Protocol
         for providing the Connectionless-mode Network Service and
         Provision of Underlying Service", International Organization for
         Standardization", International Standard 8473, May 1987.

  [10]   "End System to Intermediate System Routing Exchange Protocol for
         Use in Conjunction with the Protocol for the Provision of the
         Connectionless-mode Network Service (ISO 8473)", International
         Draft Proposal 9542.

  [11]   Kille, S., "A String Encoding of Presentation Address", Research
         Note RN/89/14, Department of Computer Science, University College
         London, February 1989.

8.   Security Considerations

     Security issues are not discussed in this memo.

9.   Author's Address:

     Greg Satz
     cisco Systems, Inc.
     1350 Willow Road
     Menlo Park, CA 94025

     Phone: (415) 326-1941

     Email: Satz@CISCO.COM