

Network Working Group  
Request for Comments: 1742  
Obsoletes: 1243  
Category: Standards Track

S. Waldbusser  
Carnegie Mellon University  
K. Frisa  
FORE Systems, Inc.  
January 1995

## AppleTalk Management Information Base II

### Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

### Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP-based internets. In particular, it defines objects for managing AppleTalk networks.

RFC 1243 defines a set of MIB objects for managing the lower layers of the AppleTalk protocol stack, up to the Network layer. This memo defines additional objects that exist in the AppleTalk portion of the MIB. These objects provide for the management of the transport and session layers of the AppleTalk protocol stack, as well as extensions to the lower layers. This is achieved in an upwardly-compatible fashion.

### Table of Contents

1. The Network Management Framework .....	2
2. Additions and Changes .....	3
2.1 New Groups .....	3
2.2 Additional Variables .....	3
2.2.1 AARP Additions .....	3
2.2.2 ATPort Additions .....	3
2.2.3 DDP Addition .....	3
2.2.4 RTMP Additions .....	4
2.2.5 KIP Addition .....	4
2.2.6 ZIP Additions .....	4
2.2.7 NBP Additions .....	4
2.2.8 ATEcho Additions .....	4
2.3 Deprecations .....	4
2.4 Changes .....	5
3. Objects .....	6

3.1 Format of Definitions .....	6
4. Overview .....	6
4.1 Structure of MIB .....	7
4.2 The LocalTalk Link Access Protocol Group .....	7
4.3 The AppleTalk Address Resolution Protocol Group .....	7
4.4 The AppleTalk Port Group .....	8
4.5 The Datagram Delivery Protocol Group .....	8
4.6 The Datagram Delivery Protocol Router Group .....	8
4.7 The Routing Table Maintenance Protocol Group .....	8
4.8 The Routing Table Maintenance Protocol Stub Group .....	8
4.9 The Kinetics Internet Protocol Group .....	8
4.10 The Zone Information Protocol Router Group .....	9
4.11 The Zone Information Protocol End Node Group .....	9
4.12 The Name Binding Protocol Group .....	9
4.13 The AppleTalk Echo Protocol Group .....	9
4.14 The AppleTalk Transaction Protocol Group .....	9
4.15 The Printer Access Protocol Group .....	9
4.16 The AppleTalk Session Protocol Group .....	9
4.17 The AppleTalk Data Stream Protocol Group .....	10
4.18 The AppleTalk Port Point to Point Group .....	10
4.19 The Per Port Counters Group .....	10
4.20 Textual Conventions .....	10
5. Definitions .....	11
6. Acknowledgments .....	82
7. References .....	83
8. Security Considerations .....	84
9. Authors' Addresses .....	84

## 1. The Network Management Framework

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

STD 16/RFC 1155 which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management.

STD 16/RFC 1212 defines a more concise description mechanism, which is wholly consistent with the SMI.

RFC 1156 which defines MIB-I, the core set of managed objects for the Internet suite of protocols. STD 17/RFC 1213 defines MIB-II, an evolution of MIB-I based on implementation experience and new operational requirements.

STD 15/RFC 1157 which defines the SNMP, the protocol used for network access to managed objects.

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

## 2. Additions and Changes

This MIB includes additions and changes to RFC 1243. These changes are outlined in the following sections.

### 2.1. New Groups

The following groups are introduced in this MIB:

- DDP Router
- RTMP Stub
- ZIP Router
- ATP
- PAP
- ASP
- ADSP
- ATPortPtoP
- Per Port Counters

### 2.2. Additional Variables

Many variables, mostly counters, were added to groups that existed in RFC 1243. These variables are listed in the following sections.

#### 2.2.1. AARP Additions

arpStatus  
arpLookups  
arpHits

#### 2.2.2. ATPort Additions

atportNetFrom  
atportZoneFrom  
atportInPkts  
atportOutPkts  
atportHome  
atportCurrentZone  
atportConflictPhysAddr  
atportZoneTable

#### 2.2.3. DDP Addition

ddpListenerTable

#### 2.2.4. RTMP Additions

- rtmpInDataPkts
- rtmpOutDataPkts
- rtmpInRequestPkts
- rtmpNextIREqualChanges
- rtmpNextIRLessChanges
- rtmpRouteDeletes
- rtmpRoutingTableOverflows

#### 2.2.5. KIP Addition

- kipFrom

#### 2.2.6. ZIP Additions

- zipNetInfoTable
- zipInErrors

#### 2.2.7. NBP Additions

- nbpAddress
- nbpSocket
- nbpEnumerator
- nbpInLookUpRequests
- nbpInLookUpReplies
- nbpInBroadcastRequests
- nbpInForwardRequests
- nbpOutLookUpReplies
- nbpRegistrationFailures
- nbpInErrors

#### 2.2.8. ATEcho Additions

- atechoOutRequests
- atechoInReplies

#### 2.3. Deprecations

The following variables have been deprecated in this version of the MIB:

- llapInPkts
- llapOutPkts
- llapInNoHandlers
- llapInErrors

These llap variables were duplicated in the interfaces table of MIB-II.

#### 2.4. Changes

The IMPORTS list has been updated to reflect the current SNMP documents.

New textual conventions have been defined.

Hyphens have been removed from enumeration strings.

Variables used as INDEXes to new tables have ACCESS not-accessible. This is because the values of the INDEX variables are contained in the object identifier for any of the other variables in the table; therefore, it does not need to be explicitly available as data.

The atportNetConfig and atportZoneConfig variables have been changed from read-only to read-write.

The atportZone variable has been renamed to atportZoneDefault, and its DESCRIPTION clause has been clarified.

The atportType, atportStatus, and kipType variables have had more values added to their enumeration lists.

The DDP group has been split into two groups; one includes variables that any AppleTalk node would implement and the other includes variables only a router would implement.

The rtmpState variable now includes another enumeration, invalid(5), which is used when deleting rows.

The variables rtmpRangeStart, rtmpRangeEnd, rtmpNextHop, rtmpType, rtmpPort, and rtmpHops have been changed from read-write to read-only.

The ZIP Group has been renamed the ZIP End Node Group.

The DESCRIPTION clause for zipZoneIndex has been clarified.

The variables zipZoneName, zipZoneNetStart, and zipZoneNetEnd have been changed from read-write to read-only.

The nbpIndex variable has been changed from read-only to read-write.

The nbpObject, nbpType, and nbpZone variables now suggest that the agent reregister its service when any of these variables is changed.

The nbpState variable includes new enumerations.

### 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.

The 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

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

### 4. Overview

AppleTalk is a protocol suite which features an open peer-to-peer architecture that runs over a variety of transmission media. AppleTalk is defined in [10]. This protocol suite interoperates with the IP protocol suite through various encapsulation methods. As large AppleTalk networks are built that coexist with large IP networks, a method to manage the AppleTalk networks with SNMP becomes necessary. This MIB defines managed objects to be used for managing AppleTalk networks.

#### 4.1. Structure of MIB

The objects are arranged into the following groups:

- LLAP
- AARP
- ATPort
- DDP
- DDP Router
- RTMP
- RTMP Stub
- KIP
- ZIP Router
- ZIP End Node
- NBP
- ATEcho
- ATP
- PAP
- ASP
- ADSP
- ATPortPtoP
- Per Port Counters

These groups are the basic unit of conformance. If the semantics of a group is applicable to an implementation, then it must implement all objects in that group. For example, a managed agent must implement the KIP group if and only if it implements the KIP protocol.

These groups are defined to provide a method for managed agents to know which objects they must implement.

#### 4.2. The LocalTalk Link Access Protocol Group

The LocalTalk Link Access Protocol (LLAP) is a medium-speed data-link protocol designed for low cost and plug-and-play operation. The LLAP group is designed to manage all interfaces on a managed device that use this protocol.

#### 4.3. The AppleTalk Address Resolution Protocol Group

The AppleTalk Address Resolution Protocol (AARP) is used to map between AppleTalk node addresses, used by the Datagram Delivery Protocol, and the addresses of the underlying data link layer. The AARP table allows for management of the Address Mapping Table on the managed device.

#### 4.4. The AppleTalk Port Group

An AppleTalk Port is a logical connection to a network over which AppleTalk packets can be transmitted. The "network" could be a tunnel, backbone network, point-to-point link, etc, as well as a native AppleTalk network. This group allows the management of the configuration of these AppleTalk ports.

#### 4.5. The Datagram Delivery Protocol Group

The Datagram Delivery Protocol (DDP) is the network-layer protocol that is responsible for the socket-to-socket delivery of datagrams over the AppleTalk Internet. This group manages the DDP layer on the managed device.

The DDP group contains statistical counters for the DDP protocol, and a table describing the DDP sockets that have protocol handlers registered.

#### 4.6. The Datagram Delivery Protocol Router Group

Some variables relevant to the Datagram Delivery Protocol (DDP) are only applicable to AppleTalk routers. These variables are included in this group.

#### 4.7. The Routing Table Maintenance Protocol Group

The Routing Table Maintenance Protocol (RTMP) is used by AppleTalk routers to create and maintain the routing tables that dictate the process of forwarding datagrams on the AppleTalk internet. The RTMP group manages the RTMP protocol as well as the routing tables generated by this protocol.

#### 4.8. The Routing Table Maintenance Protocol Stub Group

The RTMP Stub process is implemented by end nodes in order to maintain information about the routers on their networks. The variables in this group apply to both routers and end nodes. This group manages the RTMP stub process.

#### 4.9. The Kinetics Internet Protocol Group

The Kinetics Internet Protocol (KIP) is a protocol for encapsulating and routing AppleTalk datagrams over an IP internet. This name is historical. The KIP group manages the KIP routing protocol as well as the routing tables generated by this protocol.



#### 4.10. The Zone Information Protocol Router Group

The Zone Information Protocol (ZIP) is used to maintain a mapping between networks and zone names to facilitate the name lookup process performed by the Name Binding Protocol. Some variables relevant to the Zone Information Protocol (ZIP) are only applicable to AppleTalk routers. These variables are included in this group.

#### 4.11. The Zone Information Protocol End Node Group

The ZIP End Node group manages the variables relevant to the Zone Information Protocol (ZIP) that are applicable to both routers and end nodes.

#### 4.12. The Name Binding Protocol Group

The Name Binding Protocol (NBP) is a transport-level protocol that is used to convert human readable service names into the numeric AppleTalk network addresses needed for communicating across the AppleTalk network. The NBP group manages this protocol and the NBP services that exist on the managed device.

#### 4.13. The AppleTalk Echo Protocol Group

The AppleTalk Echo Protocol is a transport-level protocol used to test and verify the status of the AppleTalk internet. The AtEcho group manages this protocol.

#### 4.14. The AppleTalk Transaction Protocol Group

The AppleTalk Transaction Protocol (ATP) is a transport-level protocol that is defined to support transaction based communications. The ATP group manages this protocol.

#### 4.15. The Printer Access Protocol Group

The Printer Access Protocol (PAP) is a session-level protocol that enables communications between workstations and print servers. The PAP group manages this protocol.

#### 4.16. The AppleTalk Session Protocol Group

The AppleTalk Session Protocol (ASP) is a session-level protocol that enables sequences of communications to occur. ASP uses the services of the AppleTalk Transaction Protocol (ATP), but extends these services into the session layer. The ASP group manages this protocol.

#### 4.17. The AppleTalk Data Stream Protocol Group

The AppleTalk Data Stream Protocol (ADSP) is a session-level protocol that provides symmetric, connection-oriented, full-duplex communication between two sockets on the AppleTalk internet. In addition, ADSP handles flow-control and reliability. The ADSP group manages this protocol.

#### 4.18. The AppleTalk Port Point to Point Group

The AppleTalk Port Point to Point Group manages ports that have one or more associated point-to-point connections.

#### 4.19. The Per Port Counters Group

The Per Port Counters Group contains a set of counters which are deemed useful on a per port basis.

#### 4.20. Textual Conventions

New data types are introduced as textual conventions in this MIB document. These textual conventions enhance the readability of the specification and can ease comparison with other specifications if appropriate. It should be noted that the introduction of these textual conventions has no effect on either the syntax or the semantics of any managed objects. The use of this is merely an artifact of the explanatory method used. Objects defined in terms of this method are always encoded by means of the rules that define the primitive type. Hence, no changes to the SMI or the SNMP are necessary to accommodate these textual conventions which are adopted merely for the convenience of readers and writers in pursuit of the elusive goal of clear, concise, and unambiguous MIB documents.

The new data types are:

```

ATNetworkNumber ::=                -- 2 octets of network
                                     -- number in network
                                     -- byte order
OCTET STRING (SIZE (2))

DdpNodeAddress ::=                 -- 2 octets of net number
                                     -- in network byte order,
                                     -- 1 octet of node number
OCTET STRING (SIZE (3))

DdpSocketAddress ::=              -- 2 octets of net number
                                     -- in network byte order,
                                     -- 1 octet of node number,
```

```

-- 1 octet of socket
-- number (0..255)
OCTET STRING (SIZE (4))

ATName ::=
-- 0 to 32 octets of
-- AppleTalk ASCII [10]
OCTET STRING (SIZE (0..32))
```

## 5. Definitions

```
APPLETALK-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

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

```
-- This MIB module uses the extended OBJECT-TYPE macro as
-- defined in RFC-1212.
```

```
-- The following reference is used in this MIB:
-- [Inside AppleTalk]
-- This refers to Gursharan S. Sidhu, Richard F. Andrews, and
-- Alan B. Oppenheimer, Inside AppleTalk, Second Edition,
-- Addison Wesley, (1990).
```

```
-- AppleTalk MIB
```

```
appletalk OBJECT IDENTIFIER ::= { mib-2 13 }
```

```
ATNetworkNumber ::=
-- 2 octets of net number
-- in network byte order
OCTET STRING (SIZE (2))
```

```
DdpNodeAddress ::=
-- 2 octets of net number
-- in network byte order,
-- 1 octet of node number
OCTET STRING (SIZE (3))
```

```
DdpSocketAddress ::=
-- 2 octets of net number
-- in network byte order,
-- 1 octet of node number,
```

```

-- 1 octet of socket number
-- (0..255)
OCTET STRING (SIZE (4))

ATName ::= -- 0 to 32 octets of AppleTalk
-- ASCII [Inside AppleTalk]
OCTET STRING (SIZE (0..32))

llap          OBJECT IDENTIFIER ::= { appletalk 1 }
aarp          OBJECT IDENTIFIER ::= { appletalk 2 }
atport        OBJECT IDENTIFIER ::= { appletalk 3 }
ddp           OBJECT IDENTIFIER ::= { appletalk 4 }
rtmp          OBJECT IDENTIFIER ::= { appletalk 5 }
kip           OBJECT IDENTIFIER ::= { appletalk 6 }
zipRouter     OBJECT IDENTIFIER ::= { appletalk 7 }
nbp           OBJECT IDENTIFIER ::= { appletalk 8 }
atecho        OBJECT IDENTIFIER ::= { appletalk 9 }
atp           OBJECT IDENTIFIER ::= { appletalk 10 }
pap           OBJECT IDENTIFIER ::= { appletalk 11 }
asp           OBJECT IDENTIFIER ::= { appletalk 12 }
adsp          OBJECT IDENTIFIER ::= { appletalk 13 }
atportptop    OBJECT IDENTIFIER ::= { appletalk 14 }
rtmpStub      OBJECT IDENTIFIER ::= { appletalk 16 }
zipEndNode    OBJECT IDENTIFIER ::= { appletalk 17 }
perPort       OBJECT IDENTIFIER ::= { appletalk 18 }

-- The LLAP Group
--
-- Implementation of this group is mandatory for all
-- entities that implement LLAP
--
-- Notes for the interfaces group
--
-- When implementing the Interfaces Group of MIB-II, it is
-- suggested that the following values be used for any
-- LocalTalk interfaces:
--   ifMtu: 600
--   ifSpeed: 230000
--   ifPhysAddress: the one octet node number for the
--                   particular interface
--
-- Note also that LLAP control packets should not be
-- included in the Interfaces Group packet or octet
-- counters.
```

```

llapTable OBJECT-TYPE
    SYNTAX SEQUENCE OF LlapEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The list of LLAP entries."
    ::= { llap 1 }

llapEntry OBJECT-TYPE
    SYNTAX LlapEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "An LLAP entry containing objects for the LocalTalk
        Link Access Protocol for a particular LocalTalk
        interface.

        As an example, an instance of the llapOutPkts object
        might be named llapOutPkts.1"
    INDEX { llapIfIndex }
    ::= { llapTable 1 }

LlapEntry ::= SEQUENCE {
    llapIfIndex          INTEGER,
    llapInPkts           Counter,
    llapOutPkts          Counter,
    llapInNoHandlers     Counter,
    llapInLengthErrors   Counter,
    llapInErrors         Counter,
    llapCollisions       Counter,
    llapDefers           Counter,
    llapNoDataErrors     Counter,
    llapRandomCTSErrors  Counter,
    llapFCSErrors        Counter
}

llapIfIndex OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The LLAP interface to which this entry pertains.
        The interface identified by a particular value of
        this index is the same interface as identified
        by the same value of ifIndex."
    ::= { llapEntry 1 }

```

```
-- this object has been deprecated because it duplicates the
-- sum of the MIB-II variables ifInUcastPkts and
-- ifInNUcastPkts
```

```
llapInPkts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS deprecated
    DESCRIPTION
        "The total number of good data packets received on
        this LocalTalk interface."
    ::= { llapEntry 2 }
```

```
-- this object has been deprecated because it duplicates the
-- sum of the MIB-II variables ifOutUcastPkts and
-- ifOutNUcastPkts
```

```
llapOutPkts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS deprecated
    DESCRIPTION
        "The total number of data packets transmitted on
        this LocalTalk interface."
    ::= { llapEntry 3 }
```

```
-- this object has been deprecated because it duplicates the
-- MIB-II variable ifInUnknownProtos
```

```
llapInNoHandlers OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS deprecated
    DESCRIPTION
        "The total number of good packets received on this
        LocalTalk interface for which there was no protocol
        handler."
    ::= { llapEntry 4 }
```

```
llapInLengthErrors OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of packets received on this LocalTalk
        interface whose actual length did not match the length
        in the header."
    ::= { llapEntry 5 }
```

-- this object has been deprecated because it duplicates the  
-- MIB-II variable ifInErrors

llapInErrors OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS deprecated  
DESCRIPTION  
    "The total number of packets containing errors received  
    on this LocalTalk interface."  
::= { llapEntry 6 }

llapCollisions OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The total number of collisions assumed on this  
    LocalTalk interface due to the lack of a lapCTS reply."  
::= { llapEntry 7 }

llapDefers OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The total number of times this LocalTalk interface  
    deferred to other packets."  
::= { llapEntry 8 }

llapNoDataErrors OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The total number of times this LocalTalk interface  
    received a lapRTS packet and expected a data packet,  
    but did not receive any data packet."  
::= { llapEntry 9 }

llapRandomCTSErrors OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The total number of times this LocalTalk interface  
    received a lapCTS packet that was not solicited by a  
    lapRTS packet."

```

 ::= { llapEntry 10 }

llapFCSErrors OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of times this LocalTalk interface
        received a packet with an FCS (Frame Check Sequence)
        error."
    ::= { llapEntry 11 }

-- The AARP Group
--
-- Implementation of this group is mandatory for all entities
-- that implement AARP

aarpTable OBJECT-TYPE
    SYNTAX SEQUENCE OF AarpEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The AppleTalk Address Translation Table contains an
        equivalence of AppleTalk Network Addresses to the link
        layer physical address."
    ::= { aarp 1 }

aarpEntry OBJECT-TYPE
    SYNTAX AarpEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "Each entry contains one AppleTalk Network Address to
        physical address equivalence.

        As an example, an instance of the aarpPhysAddress
        object might be named aarpPhysAddress.1.0.80.234"
    INDEX { aarpIfIndex, aarpNetAddress }
    ::= { aarpTable 1 }

AarpEntry ::= SEQUENCE {
    aarpIfIndex      INTEGER,
    aarpPhysAddress  OCTET STRING,
    aarpNetAddress   DdpNodeAddress,
    aarpStatus       INTEGER
}

```



**aarpIfIndex OBJECT-TYPE**

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

"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."

::= { aarpEntry 1 }

**aarpPhysAddress OBJECT-TYPE**

SYNTAX OCTET STRING

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The media-dependent physical address."

::= { aarpEntry 2 }

**aarpNetAddress OBJECT-TYPE**

SYNTAX DdpNodeAddress

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The AppleTalk Network Address corresponding to the media-dependent physical address."

::= { aarpEntry 3 }

**aarpStatus OBJECT-TYPE**

SYNTAX INTEGER {

valid(1),

invalid(2)

}

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The status of this AARP entry.

Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the aarpTable. That is, it effectively disassociates 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 from agents tabular information corresponding to entries not currently in use. Proper interpretation of such entries requires examination of the relevant aarpStatus object."

```
 ::= { aarpEntry 4 }

aarpLookups OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times the AARP cache for this entity
         was searched."
    ::= { aarp 2 }

aarpHits OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times an entry was searched for and
         found in the AARP cache for this entity."
    ::= { aarp 3 }

-- The ATPort Group
--
-- Implementation of this group is mandatory for all entities
-- that implement AppleTalk ports
--
-- Note that to be compliant with this group, all variables
-- that have read-write access must be implemented as
-- read-write.

atportTable OBJECT-TYPE
    SYNTAX SEQUENCE OF AtportEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A list of AppleTalk ports for this entity."
    ::= { atport 1 }

atportEntry OBJECT-TYPE
    SYNTAX AtportEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The description of one of the AppleTalk
         ports on this entity.

        As an example, an instance of the atportNetFrom object
        might be named atportNetFrom.2"
```

```

INDEX { atportIndex }
 ::= { atportTable 1 }

AtportEntry ::= SEQUENCE {
    atportIndex                INTEGER,
    atportDescr                DisplayString,
    atportType                 INTEGER,
    atportNetStart              ATNetworkNumber,
    atportNetEnd                ATNetworkNumber,
    atportNetAddress            DdpNodeAddress,
    atportStatus                INTEGER,
    atportNetConfig             INTEGER,
    atportZoneConfig            INTEGER,
    atportZoneDefault           ATName,
    atportIfIndex               INTEGER,
    atportNetFrom               DdpNodeAddress,
    atportZoneFrom              DdpNodeAddress,
    atportInPkts                Counter,
    atportOutPkts               Counter,
    atportHome                  INTEGER,
    atportCurrentZone           ATName,
    atportConflictPhysAddr      OCTET STRING
}

atportIndex OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "A unique value for each AppleTalk port.
        Its value is between 1 and the total number of
        AppleTalk ports. The value for each port must
        remain constant at least from the re-initialization
        of the entity's network management system to the
        next re-initialization."
    ::= { atportEntry 1 }

atportDescr OBJECT-TYPE
    SYNTAX DisplayString
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "A text string containing information about the
        port. This string is intended for presentation
        to a human; it must not contain anything but printable
        ASCII characters."
    ::= { atportEntry 2 }

```

```
-- Several objects throughout the MIB key off of atportType to
-- determine the format of OCTET STRING addresses of peers.
-- The address formats are as follows:
--   localtalk, ethertalk1, ethertalk2, tokentalk, iptalk,
--   fdditalk, smdstalk, arctalk, and virtual take the
--   format of DdpNodeAddress
--   serialPPP: null OCTET STRING
--   serialNonstandard: vendor specific
--   aurp: see AURP MIB to determine format
--   frameRelay: 32 bit DLCI in network byte order
--   (OCTET STRING (SIZE (4)))
--   x25: X121Address (see RFC 1382)
--   ip: IP address (OCTET STRING (SIZE (4)))
--   osi: NSAP (OCTET STRING (SIZE (3..20)))
--   decnetIV: 6 bit area, 10 bit host in network byte order
--   (OCTET STRING (SIZE (2)))
--   arap: ???
--   nonAppleTalk3Com: based on ifType
--   ipx: 32 bit network number in network byte order
--   followed by datalink address of host
--   arns: 32 bit ARNS header
--   hdlc: DdpNodeAddress or null OCTET STRING
```

atportType OBJECT-TYPE

```
SYNTAX INTEGER {
    other(1),          -- none of the following
    localtalk(2),
    ethertalk1(3),
    ethertalk2(4),
    tokentalk(5),
    iptalk(6),
    serialPPP(7),
    serialNonstandard(8),
    virtual(9),        -- an internal interface
    fdditalk(10),
    arctalk(11),
    smdstalk(12),
    aurp(13),
    frameRelay(14),
    x25(15),
    ip(16),
    osi(17),
    decnetIV(18),
    arap(19),
    isdnInThePacketMode(20),
    nonAppleTalk3Com(21),
    ipx(22),
    arns(23),
```

```
        hdlc(24)
    }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The type of port, distinguished by the protocol
        immediately below DDP in the protocol stack."
    ::= { atportEntry 3 }

atportNetStart OBJECT-TYPE
    SYNTAX ATNetworkNumber
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The first AppleTalk network address in the range
        configured for this port.  If this port is not a
        native AppleTalk port, this object shall have the
        value of two octets of zero."
    ::= { atportEntry 4 }

atportNetEnd OBJECT-TYPE
    SYNTAX ATNetworkNumber
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The last AppleTalk network address in the range
        configured for this port.  If the network to which
        this AppleTalk port is connected is a non-extended
        network, or if it is not a native AppleTalk port,
        the value for atportNetEnd shall be two octets of
        zero."
    ::= { atportEntry 5 }

atportNetAddress OBJECT-TYPE
    SYNTAX DdpNodeAddress
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The AppleTalk network address configured for this
        port.  In addition, this value may be used as a hint
        for an initial node number used during node-finding.
        If this port is not a native AppleTalk port, this
        object shall have the value of three octets of zero."
    ::= { atportEntry 6 }

atportStatus OBJECT-TYPE
    SYNTAX INTEGER {
        routing(1), --this port is fully configured & routing
```

```

    unconfigured(2),
    off(3),
    invalid(4),
    endNode(5), -- this port is acting as an end node
    offDueToConflict(6), -- port is off due to
        -- configuration conflict
    other(7) -- none of the states defined above
}
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The configuration status of this port.

    Setting this object to the value invalid(4) has the
    effect of invalidating the corresponding entry in the
    atportTable. That is, it effectively disassociates 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 from agents tabular information corresponding
    to entries not currently in use. Proper
    interpretation of such entries requires examination
    of the relevant atportStatus object."
::= { atportEntry 7 }

```

#### atportNetConfig OBJECT-TYPE

```

SYNTAX INTEGER {
    conflictOrientedSeed(1), -- use configured network
        -- range even if it conflicts with another
        -- AppleTalk device
    garnered(2), -- acquire from another AppleTalk device
    guessed(3), -- generate a "random" network range
    unconfigured(4), -- no other value applies
    conflictAverseSeed(5), -- use configured network
        -- range, but don't come up if it conflicts
    softSeed(6) -- attempt to use configured network
        -- range, but use network range from another
        -- router if our configuration conflicts
}
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The status of the network information for this port.
    If this port is not a native AppleTalk port, this
    object shall have the value unconfigured(4)."
::= { atportEntry 8 }

```

## atportZoneConfig OBJECT-TYPE

```
SYNTAX INTEGER {
    conflictOrientedSeed(1), -- use configured zone
    -- information even if it conflicts with
    -- another AppleTalk device
    garnered(2), -- acquire from another AppleTalk device
    guessed(3), -- generate "random" zone information
    unconfigured(4), -- no other value applies
    conflictAverseSeed(5), -- use configured zone
    -- information, but don't come up if it
    -- conflicts
    softSeed(6) -- attempt to use configured zone
    -- information, but use zone information
    -- from another router if our configuration
    -- conflicts
}
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The status of the zone information for this port.
    If this port is not a native AppleTalk port, this
    object shall have the value unconfigured(4)."
```

::= { atportEntry 9 }

## atportZoneDefault OBJECT-TYPE

```
SYNTAX ATName
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The name of the default zone for this port. If
    this port only has one zone, that zone is
    represented here. If this port is not a native
    AppleTalk port, this object shall contain an octet
    string of zero length.

    When this value is changed in a router, the router
    must send a zipNotify packet on the associated
    network."
```

::= { atportEntry 10 }

## atportIfIndex OBJECT-TYPE

```
SYNTAX INTEGER
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The physical interface associated with this
    AppleTalk port. The interface identified by a
    particular value of this index is the same interface
```

as identified by the same value of ifIndex."  
 ::= { atportEntry 11 }

atportNetFrom OBJECT-TYPE

SYNTAX DdpNodeAddress

ACCESS read-only

STATUS mandatory

DESCRIPTION

"When atportNetConfig is set to garnered(2), this variable contains the DDP address of an entity from which the AppleTalk network number was garnered. When atportNetConfig is set to conflictOrientedSeed(1), conflictAverseSeed(5), or softSeed(6), this variable contains the DDP address of an entity which confirmed or supplied our AppleTalk network number, for example by replying to a ZIP GetNetInfo request.

If atportNetConfig is set to guessed(3) or unconfigured(4), or if the entity has not received any network number confirmation, this variable should be set to three octets of zero."

::= { atportEntry 12 }

atportZoneFrom OBJECT-TYPE

SYNTAX DdpNodeAddress

ACCESS read-only

STATUS mandatory

DESCRIPTION

"When atportZoneConfig is set to garnered(2), this variable contains the DDP address of an entity from which the AppleTalk zone list was garnered.

When atportZoneConfig is set to conflictOrientedSeed(1), conflictAverseSeed(5), or softSeed(6), this variable contains the DDP address of an entity which confirmed or supplied our AppleTalk zone information, for example by replying to a ZIP GetNetInfo request or a ZIP Query.

If atportZoneConfig is set to guessed(3) or unconfigured(4), or if the entity has not received any zone confirmation, this variable should be set to three octets of zero."

::= { atportEntry 13 }



```
atportInPkts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of packets received by this entity on
        this port."
    ::= { atportEntry 14 }

atportOutPkts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of packets transmitted by this entity on
        this port."
    ::= { atportEntry 15 }

atportHome OBJECT-TYPE
    SYNTAX INTEGER {
        home(1),
        notHome(2)
    }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "An indication of whether or not the entity is
        homed on this port, that is to say, a port on which
        the entity could perform NBP registrations for
        services that it chooses to advertise."
    ::= { atportEntry 16 }

atportCurrentZone OBJECT-TYPE
    SYNTAX ATName
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The current zone for the port.  In general, this is
        the zone name in which services on this port will
        be registered.  If this port is not a native
        AppleTalk port, this object shall contain an octet
        string of zero length.  Note that modifications to
        this object do not affect the nbpTable."
    ::= { atportEntry 17 }

atportConflictPhysAddr OBJECT-TYPE
    SYNTAX OCTET STRING
    ACCESS read-only
```

```

STATUS mandatory
DESCRIPTION
    "The link-layer address of a device which caused
    this entity to set atportStatus to
    offDueToConflict(6). If this address is not
    available, or if the entity has not set atportStatus
    to offDueToConflict, this object shall be a zero
    length OCTET STRING."
 ::= { atportEntry 18 }

-- The atportZoneTable stores information about the zones
-- associated with each port. The default zone for each
-- port is stored in the port's atportZoneDefault variable;
-- all other zones for the port are listed in this table.
-- If a port only has one zone, it should be stored in the
-- port's atportZoneDefault variable, and this table should
-- be empty.
--
-- One of the indexes for this table is atportZoneName.
-- Even though AppleTalk zone name matches are
-- case-insensitive, this table will store zone names
-- regardless of case. SNMP Get, GetNext and Set operations
-- are performed on these (potentially) mixed case strings
-- according to the normal SNMP rules with the following
-- caveat: in processing a SET request, the agent shall
-- perform a case-insensitive search and a case-sensitive
-- search. If the case-insensitive search matches and the
-- case-sensitive search does not match, the "equivalent"
-- zone name exists in another entry with a different
-- capitalization and the SET request shall fail due
-- to the name being inconsistent (SNMPv1 should return a
-- genErr.) This insures that only one version of a zone
-- name will appear in each agent, at the expense of forcing
-- a management station to query using that exact name.

atportZoneTable OBJECT-TYPE
    SYNTAX SEQUENCE OF AtportZoneEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The table of zone information for non-default
        zones on ports."
    ::= { atport 2 }

atportZoneEntry OBJECT-TYPE
    SYNTAX AtportZoneEntry
    ACCESS not-accessible
    STATUS mandatory

```

## DESCRIPTION

"An entry of zone information for a port.

As an example, an instance of the atportZoneStatus object might be named

atportZoneStatus.2.8.84.119.105.108.105.103.104.116"

INDEX { atportZonePort, atportZoneName }

::= { atportZoneTable 1 }

```
AtportZoneEntry ::= SEQUENCE {
    atportZonePort      INTEGER,
    atportZoneName      ATName (SIZE (1..32)),
    atportZoneStatus    INTEGER
}
```

## atportZonePort OBJECT-TYPE

SYNTAX INTEGER

ACCESS not-accessible

STATUS mandatory

## DESCRIPTION

"An integer representing the port to which this zone belongs. The port identified by a particular value of this object is the same port as identified by the same value of atportIndex."

::= { atportZoneEntry 1 }

## atportZoneName OBJECT-TYPE

SYNTAX ATName (SIZE (1..32))

ACCESS not-accessible

STATUS mandatory

## DESCRIPTION

"A zone name configured for the AppleTalk port referred to in the corresponding entry of atportZonePort.

When this value is changed in a router, the router must send a zipNotify packet on the associated network."

::= { atportZoneEntry 2 }

## atportZoneStatus OBJECT-TYPE

```
SYNTAX INTEGER {
    valid(1),
    invalid(2)
}
```

ACCESS read-write

STATUS mandatory

## DESCRIPTION

"The status of this zone entry.

Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the atportZoneTable. That is, it effectively disassociates 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 from agents tabular information corresponding to entries not currently in use. Proper interpretation of such entries requires examination of the relevant atportZoneStatus object."

```
::= { atportZoneEntry 3 }
```

```
-- The DDP Group
```

```
--
```

```
-- Implementation of this group is mandatory for all
-- entities that implement DDP
```

```
--
```

```
-- This group consists of DDP variables that would be
-- implemented by either a router or an end node. The
-- following variables are included:
```

```
-- ddpOutRequests
-- ddpOutShorts
-- ddpOutLongs
-- ddpInReceives
-- ddpInLocalDatagrams
-- ddpNoProtocolHandlers
-- ddpTooShortErrors
-- ddpTooLongErrors
-- ddpShortDDPErrors
-- ddpChecksumErrors
-- ddpListenerTable
--
```

```
-- Note that the variables in this group are not numbered
-- sequentially. This was done so that it was not necessary
-- to deprecate variables from RFC 1243.
```

```
ddpOutRequests OBJECT-TYPE
```

```
SYNTAX Counter
```

```
ACCESS read-only
```

```
STATUS mandatory
```

```
DESCRIPTION
```

```
"The total number of DDP datagrams which were
supplied to DDP by local DDP clients in requests for
```

transmission. Note that this counter does not include any datagrams counted in ddpForwRequests."  
 ::= { ddp 1 }

ddpOutShorts OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The total number of short DDP datagrams which were transmitted from this entity."

::= { ddp 2 }

ddpOutLongs OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The total number of long DDP datagrams which were transmitted from this entity."

::= { ddp 3 }

ddpInReceives OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The total number of input datagrams received by DDP, including those received in error."

::= { ddp 4 }

ddpInLocalDatagrams OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The total number of input DDP datagrams for which this entity was their final DDP destination."

::= { ddp 6 }

ddpNoProtocolHandlers OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The total number of DDP datagrams addressed to this entity that were addressed to an upper layer protocol"

for which no protocol handler existed."  
 ::= { ddp 7 }

ddpTooShortErrors OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The total number of input DDP datagrams dropped because the received data length was less than the data length specified in the DDP header or the received data length was less than the length of the expected DDP header."

::= { ddp 9 }

ddpTooLongErrors OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The total number of input DDP datagrams dropped because they exceeded the maximum DDP datagram size."

::= { ddp 10 }

ddpShortDDPErrors OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The total number of input DDP datagrams dropped because this entity was not their final destination and their type was short DDP."

::= { ddp 12 }

ddpChecksumErrors OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The total number of input DDP datagrams for which this DDP entity was their final destination, and which were dropped because of a checksum error."

::= { ddp 14 }

ddpListenerTable OBJECT-TYPE

SYNTAX SEQUENCE OF DdpListenerEntry  
ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"The ddpListenerTable stores information for each DDP socket that has a listener."

::= { ddp 15 }

ddpListenerEntry OBJECT-TYPE

SYNTAX DdpListenerEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"This ddpListenerEntry contains information about a particular socket that has a socket listener."

As an example, an instance of the ddpListenerStatus object might be named ddpListenerStatus.0.80.220.1"

INDEX { ddpListenerAddress }

::= { ddpListenerTable 1 }

DdpListenerEntry ::= SEQUENCE {

ddpListenerAddress DdpSocketAddress,

ddpListenerInPkts Counter,

ddpListenerStatus INTEGER

}

ddpListenerAddress OBJECT-TYPE

SYNTAX DdpSocketAddress

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"The DDP address that this socket listener is bound to. If this socket listener isn't bound to a particular address, for instance if it is intended for all interfaces, this object shall have the value of three octets of zero followed by one octet of socket number. The socket number must not equal zero."

::= { ddpListenerEntry 1 }

ddpListenerInPkts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of packets received for this listener."

::= { ddpListenerEntry 2 }

## ddpListenerStatus OBJECT-TYPE

```
SYNTAX INTEGER {
    valid(1),
    invalid(2)
}
```

```
ACCESS read-write
```

```
STATUS mandatory
```

## DESCRIPTION

"The status of this socket listener. Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the ddpListenerTable. That is, it effectively disassociates 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 from agents tabular information corresponding to entries not currently in use. Proper interpretation of such entries requires examination of the relevant ddpListenerStatus object."

```
::= { ddpListenerEntry 3 }
```

```
-- The DDP Router Group
```

```
--
```

```
-- Implementation of this group is required for all routers
```

```
-- which implement DDP
```

```
--
```

```
-- This group consists of DDP variables that only a router  
-- would implement. The following variables are included:
```

```
--     ddpForwRequests
```

```
--     ddpOutNoRoutes
```

```
--     ddpBroadcastErrors
```

```
--     ddpHopCountErrors
```

```
--     ddpForwardingTable
```

```
--
```

```
-- Note that the variables in this group are not numbered  
-- sequentially. This was done so that variables from
```

```
-- RFC 1243 did not need to be deprecated.
```

## ddpForwRequests OBJECT-TYPE

```
SYNTAX Counter
```

```
ACCESS read-only
```

```
STATUS mandatory
```

## DESCRIPTION

"The number of input datagrams for which this entity was not their final DDP destination, as a result of



which an attempt was made to find a route to forward them to that final destination."  
 ::= { ddp 5 }

ddpOutNoRoutes OBJECT-TYPE

SYNTAX Counter  
 ACCESS read-only  
 STATUS mandatory  
 DESCRIPTION

"The total number of DDP datagrams dropped because a route could not be found to their final destination."  
 ::= { ddp 8 }

ddpBroadcastErrors OBJECT-TYPE

SYNTAX Counter  
 ACCESS read-only  
 STATUS mandatory  
 DESCRIPTION

"The total number of input DDP datagrams dropped because this entity was not their final destination and they were addressed to the link level broadcast."  
 ::= { ddp 11 }

ddpHopCountErrors OBJECT-TYPE

SYNTAX Counter  
 ACCESS read-only  
 STATUS mandatory  
 DESCRIPTION

"The total number of input DDP datagrams dropped because this entity was not their final destination and their hop count would exceed 15."  
 ::= { ddp 13 }

-- The ddpForwardingTable is a read-only table which shows the  
 -- next hop that a datagram will take when being routed to a  
 -- specific network. If a manager wishes to change data in  
 -- this table via SNMP, he must change it in the MIB for the  
 -- routing protocol itself (by incrementing hop counts,  
 -- etc), rather than in this table. This table is derived  
 -- by the managed entity from the information it receives  
 -- from the routing protocols that it supports.  
 --

-- This table also shows the routing table from which the next  
 -- hop was derived. When a MIB is written for an AppleTalk  
 -- routing protocol, it should include the definition of an  
 -- object identifier which will be used in the  
 -- ddpForwardingProto variable defined here. (For example,  
 -- a value for RTMP is defined as { ddp-forw-proto-oids 1 }

```
-- below.)
--
-- To look for a specific net N in this table, it is suggested
-- that the management station perform a get-next query for
-- ddpForwardingNetEnd.(N-1). This will retrieve the correct
-- row if it exists in the table.
```

```
ddpForwardingTable OBJECT-TYPE
    SYNTAX SEQUENCE OF DdpForwardingEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A table of forwarding entries for DDP. This table
        contains a route for each AppleTalk network currently
        known to the entity."
    ::= { ddp 16 }
```

```
ddpForwardingEntry OBJECT-TYPE
    SYNTAX DdpForwardingEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A forwarding entry for a particular AppleTalk
        network.

        As an example, an instance of the ddpForwardingPort
        object might be named ddpForwardingPort.0.90"
    INDEX { ddpForwardingNetEnd }
    ::= { ddpForwardingTable 1 }
```

```
DdpForwardingEntry ::= SEQUENCE {
    ddpForwardingNetEnd      ATNetworkNumber,
    ddpForwardingNetStart    ATNetworkNumber,
    ddpForwardingNextHop     OCTET STRING,
    ddpForwardingProto       OBJECT IDENTIFIER,
    ddpForwardingModifiedTime TimeTicks,
    ddpForwardingUseCounts    Counter,
    ddpForwardingPort        INTEGER
}
```

```
ddpForwardingNetEnd OBJECT-TYPE
    SYNTAX ATNetworkNumber
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The last network number in the network range
        matched by this forwarding entry. This will not be
        zero even if this corresponds to a non-extended
```

```
        net."
 ::= { ddpForwardingEntry 1 }

ddpForwardingNetStart OBJECT-TYPE
    SYNTAX ATNetworkNumber
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The first network number in the network range
        matched by this forwarding entry."
 ::= { ddpForwardingEntry 2 }

ddpForwardingNextHop OBJECT-TYPE
    SYNTAX OCTET STRING
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The next hop in the route to this entry's
        destination network. The format of this address can
        be determined by examining the atportType
        corresponding to this entry."
 ::= { ddpForwardingEntry 3 }

ddpForwardingProto OBJECT-TYPE
    SYNTAX OBJECT IDENTIFIER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The routing mechanism by which this route was
        learned."
 ::= { ddpForwardingEntry 4 }

ddpForwardingModifiedTime OBJECT-TYPE
    SYNTAX TimeTicks
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The value of sysUpTime at the time of the last
        modification to this entry. The initial value of
        ddpForwardingModified time shall be the value of
        sysUpTime at the time the entry is created."
 ::= { ddpForwardingEntry 5 }

ddpForwardingUseCounts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
```

```

        "The number of times this entry has been used to
        route a packet to the destination network. Note
        that this counter is not cleared when the
        corresponding ddpForwardingNextHop variable
        changes."
 ::= { ddpForwardingEntry 6 }

ddpForwardingPort OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The AppleTalk port through which
        ddpForwardingNextHop is reached. The interface
        identified by a particular value of this variable is
        the same interface as identified by the same value
        of atportIndex."
 ::= { ddpForwardingEntry 7 }

ddpForwProtoOids OBJECT IDENTIFIER ::= { ddp 17 }

-- The value to be assigned to ddpForwardingProto when the
-- routing protocol is RTMP.
rtmpRoutingProto OBJECT IDENTIFIER ::= { ddpForwProtoOids 1 }

-- The value to be assigned to ddpForwardingProto when the
-- routing protocol is KIP.
kipRoutingProto OBJECT IDENTIFIER ::= { ddpForwProtoOids 2 }

ddpForwardingTableOverflows OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times the entity attempted to add an
        entry to the forwarding table but failed due to
        overflow."
 ::= { ddp 18 }

-- The RTMP Group
--
-- Implementation of this group is required for all routers
-- which implement RTMP

rtmpTable OBJECT-TYPE
    SYNTAX SEQUENCE OF RtmpEntry

```

ACCESS not-accessible  
 STATUS mandatory  
 DESCRIPTION

"A list of Routing Table Maintenance Protocol  
 entries for this entity."

::= { rtmp 1 }

rtmpEntry OBJECT-TYPE

SYNTAX RtmpEntry  
 ACCESS not-accessible  
 STATUS mandatory  
 DESCRIPTION

"The route entry to a particular network range.

As an example, an instance of the rtmpPort object  
 might be named rtmpPort.0.80"

INDEX { rtmpRangeStart }  
 ::= { rtmpTable 1 }

RtmpEntry ::= SEQUENCE {  
   rtmpRangeStart ATNetworkNumber,  
   rtmpRangeEnd ATNetworkNumber,  
   rtmpNextHop OCTET STRING,  
   rtmpType INTEGER,  
   rtmpPort INTEGER,  
   rtmpHops INTEGER,  
   rtmpState INTEGER  
 }

rtmpRangeStart OBJECT-TYPE

SYNTAX ATNetworkNumber  
 ACCESS read-only  
 STATUS mandatory  
 DESCRIPTION

"The first DDP network address in the network range  
 to which this routing entry pertains. This is a two  
 octet DDP network address in network byte order."

::= { rtmpEntry 1 }

rtmpRangeEnd OBJECT-TYPE

SYNTAX ATNetworkNumber  
 ACCESS read-only  
 STATUS mandatory  
 DESCRIPTION

"The last DDP network address in the network range  
 to which this routing entry pertains. This is a two  
 octet DDP network address in network byte order. If  
 the network to which this routing entry pertains is

a non-extended network, the value for rtmpRangeEnd shall be two octets of zero."  
 ::= { rtmpEntry 2 }

rtmpNextHop OBJECT-TYPE  
SYNTAX OCTET STRING  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
"The next internet router in the route to this entry's destination network. The format of this address can be determined by examining the atportType corresponding to this entry."  
 ::= { rtmpEntry 3 }

rtmpType OBJECT-TYPE  
SYNTAX INTEGER {  
    other(1),  
    appletalk(2),  
    serialPPP(3),  
    serialNonstandard(4)  
}  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
"The type of network over which this route points."  
 ::= { rtmpEntry 4 }

rtmpPort OBJECT-TYPE  
SYNTAX INTEGER  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
"The AppleTalk port over which this route points. The interface identified by a particular value of this variable is the same interface as identified by the same value of atportIndex."  
 ::= { rtmpEntry 5 }

rtmpHops OBJECT-TYPE  
SYNTAX INTEGER  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
"The number of hops required to reach the destination network to which this routing entry pertains."  
 ::= { rtmpEntry 6 }

## rtmpState OBJECT-TYPE

```
SYNTAX INTEGER {  
    good(1),  
    suspect(2),  
    badZero(3),  
    badOne(4),  
    invalid(5)  
}
```

ACCESS read-write

STATUS mandatory

## DESCRIPTION

"The status of the information contained in this route entry.

Setting this object to the value invalid(5) has the effect of invalidating the corresponding entry in the rtmpTable. That is, it effectively disassociates 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 from agents tabular information corresponding to entries not currently in use. Proper interpretation of such entries requires examination of the relevant rtmpState object."

```
::= { rtmpEntry 7 }
```

## rtmpInDataPkts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"A count of the number of good RTMP data packets received by this entity."

```
::= { rtmp 2 }
```

## rtmpOutDataPkts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"A count of the number of RTMP packets sent by this entity."

```
::= { rtmp 3 }
```

## rtmpInRequestPkts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "A count of the number of good RTMP Request packets  
    received by this entity."  
::= { rtmp 4 }

rtmpNextIREqualChanges OBJECT-TYPE  
    SYNTAX Counter  
    ACCESS read-only  
    STATUS mandatory  
    DESCRIPTION  
        "A count of the number of times RTMP changes the  
        Next Internet Router in a routing entry because the  
        hop count advertised in a routing tuple was equal to  
        the current hop count for a particular network."  
    ::= { rtmp 5 }

rtmpNextIRLessChanges OBJECT-TYPE  
    SYNTAX Counter  
    ACCESS read-only  
    STATUS mandatory  
    DESCRIPTION  
        "A count of the number of times RTMP changes the  
        Next Internet Router in a routing entry because the  
        hop count advertised in a routing tuple was less  
        than the current hop count for a particular network."  
    ::= { rtmp 6 }

rtmpRouteDeletes OBJECT-TYPE  
    SYNTAX Counter  
    ACCESS read-only  
    STATUS mandatory  
    DESCRIPTION  
        "A count of the number of times RTMP deletes a route  
        because it was aged out of the table. This can help  
        to detect routing problems."  
    ::= { rtmp 7 }

rtmpRoutingTableOverflows OBJECT-TYPE  
    SYNTAX Counter  
    ACCESS read-only  
    STATUS mandatory  
    DESCRIPTION  
        "The number of times RTMP attempted to add a route  
        to the RTMP table but failed due to lack of space."  
    ::= { rtmp 8 }



```
-- The RTMP Stub Group
--
-- Implementation of this group is mandatory for all
-- entities that implement RTMP
--
-- It is intended that this group be implemented by routers
-- and end nodes.
```

rtmpOutRequestPkts OBJECT-TYPE

```
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
    "A count of the number of RTMP Request packets sent
    by this entity."
::= { rtmpStub 1 }
```

rtmpInVersionMismatches OBJECT-TYPE

```
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
    "A count of the number of RTMP packets received by
    this entity that were rejected due to a version
    mismatch."
::= { rtmpStub 2 }
```

rtmpInErrors OBJECT-TYPE

```
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
    "A count of the number of RTMP packets received by
    this entity that were rejected for an error other
    than version mismatch."
::= { rtmpStub 3 }
```

```
-- The KIP Group
--
-- Implementation of this group is mandatory for all
-- entities that implement KIP
```

kipTable OBJECT-TYPE

```
SYNTAX SEQUENCE OF KipEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
```

"The table of routing information for KIP networks."  
 ::= { kip 1 }

kipEntry OBJECT-TYPE

SYNTAX KipEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"An entry in the routing table for KIP networks.

As an example, an instance of the kipCore object  
 might be named kipCore.0.80"

INDEX { kipNetStart }

::= { kipTable 1 }

KipEntry ::= SEQUENCE {

kipNetStart ATNetworkNumber,

kipNetEnd ATNetworkNumber,

kipNextHop IpAddress,

kipHopCount INTEGER,

kipBCastAddr IpAddress,

kipCore INTEGER,

kipType INTEGER,

kipState INTEGER,

kipShare INTEGER,

kipFrom IpAddress

}

kipNetStart OBJECT-TYPE

SYNTAX ATNetworkNumber

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The first AppleTalk network address in the range  
 for this routing entry. This address is a two octet  
 DDP network address in network byte order."

::= { kipEntry 1 }

kipNetEnd OBJECT-TYPE

SYNTAX ATNetworkNumber

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The last AppleTalk network address in the range for  
 this routing entry. This address is a two octet DDP  
 network address in network byte order. If the  
 network to which this AppleTalk port is connected is  
 a non-extended network, the value for kipNetEnd

shall be two octets of zero."  
 ::= { kipEntry 2 }

kipNextHop OBJECT-TYPE

SYNTAX IPAddress

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The IP address of the next hop in the route to this  
entry's destination network."

::= { kipEntry 3 }

kipHopCount OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The number of hops required to reach the destination  
network to which this entry pertains."

::= { kipEntry 4 }

kipBCastAddr OBJECT-TYPE

SYNTAX IPAddress

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The form of the IP address used to broadcast on this  
network."

::= { kipEntry 5 }

kipCore OBJECT-TYPE

SYNTAX INTEGER {

core(1),

notcore(2)

}

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The status of kipNextHop as a core gateway."

::= { kipEntry 6 }

kipType OBJECT-TYPE

SYNTAX INTEGER {

kipRouter(1),

net(2),

host(3),

other(4),

async(5)

```
    }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The type of the entity that this route points to."
    ::= { kipEntry 7 }

kipState OBJECT-TYPE
    SYNTAX INTEGER {
        configured(1), -- this entry is not aged
        learned(2),
        invalid(3)
    }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The state of this network entry.

        Setting this object to the value invalid(3) has the
        effect of invalidating the corresponding entry in the
        kipTable. That is, it effectively disassociates 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 from agents tabular information corresponding
        to entries not currently in use. Proper
        interpretation of such entries requires examination
        of the relevant kipState object."
    ::= { kipEntry 8 }

kipShare OBJECT-TYPE
    SYNTAX INTEGER {
        shared(1),
        private(2)
    }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "If the information in this entry is propagated to
        other routers as part of the AA routing protocol,
        the value of this variable is equal to shared(1).
        Otherwise its value is private(2)."
    ::= { kipEntry 9 }

kipFrom OBJECT-TYPE
    SYNTAX IpAddress
    ACCESS read-only
```

```

STATUS mandatory
DESCRIPTION
    "The IP address from which the routing entry was
    learned via the AA protocol.  If this entry was not
    created via the AA protocol, it should contain IP
    address 0.0.0.0."
 ::= { kipEntry 10 }

-- The ZIP Router Group
--
-- Implementation of this group is required for all routers
-- which implement ZIP
--
-- This group consists of ZIP variables that would be
-- implemented by a router.

zipTable OBJECT-TYPE
    SYNTAX SEQUENCE OF ZipEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The table of zone information for reachable
        AppleTalk networks."
    ::= { zipRouter 1 }

zipEntry OBJECT-TYPE
    SYNTAX ZipEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "An entry of zone information for a particular zone
        and network combination.

        As an example, an instance of the zipZoneState object
        might be named zipZoneState.0.80.4"
    INDEX { zipZoneNetStart, zipZoneIndex }
    ::= { zipTable 1 }

ZipEntry ::= SEQUENCE {
    zipZoneName      ATName,
    zipZoneIndex     INTEGER,
    zipZoneNetStart  ATNetworkNumber,
    zipZoneNetEnd    ATNetworkNumber,
    zipZoneState     INTEGER,
    zipZoneFrom      OCTET STRING,
    zipZonePort      INTEGER
}

```

**zipZoneName OBJECT-TYPE**

SYNTAX ATName

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The zone name of this entry. This is stored in Mac ASCII format. If the full zone list for the entry is not known, the value for zipZoneName shall be a zero length octet string."

::= { zipEntry 1 }

**zipZoneIndex OBJECT-TYPE**

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

"An integer that is unique to the zipZoneName that is present in this entry. For any given zone name, every zipEntry that has an equal zone name will have the same zipZoneIndex. When a zone name is discovered which is not currently in the table, it will be assigned an index greater than any previously assigned index."

::= { zipEntry 2 }

**zipZoneNetStart OBJECT-TYPE**

SYNTAX ATNetworkNumber

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The network that starts the range for this entry. This address is a two octet DDP network address in network byte order."

::= { zipEntry 3 }

**zipZoneNetEnd OBJECT-TYPE**

SYNTAX ATNetworkNumber

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The network that ends the range for this entry. This address is a two octet DDP network address in network byte order. If the network to which this zip entry pertains is a non-extended network, the value for zipZoneNetEnd shall be two octets of zero."

::= { zipEntry 4 }

## zipZoneState OBJECT-TYPE

```
SYNTAX INTEGER {  
    valid(1),  
    invalid(2)  
}
```

ACCESS read-write

STATUS mandatory

## DESCRIPTION

"The state of this zip entry.

Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the zipTable. That is, it effectively disassociates 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 from agents tabular information corresponding to entries not currently in use. Proper interpretation of such entries requires examination of the relevant zipZoneState object."

```
::= { zipEntry 5 }
```

## zipZoneFrom OBJECT-TYPE

SYNTAX OCTET STRING

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"The address from which this zone name to network number mapping was learned. The format of this address can be determined by examining the atportType corresponding to this entry. When this mapping is learned from the entity itself, this object shall have the value of three octets of zero."

```
::= { zipEntry 6 }
```

## zipZonePort OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"The AppleTalk port through which this zone name to network number mapping was learned. The interface identified by a particular value of this variable is the same interface as identified by the same value of atportIndex."

```
 ::= { zipEntry 7 }

zipInZipQueries OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ZIP Queries received by this entity."
    ::= { zipRouter 2 }

zipInZipReplies OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ZIP Replies received by this entity."
    ::= { zipRouter 3 }

zipInZipExtendedReplies OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ZIP Extended Replies received by this
        entity."
    ::= { zipRouter 4 }

zipZoneConflictErrors OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times a conflict has been detected
        between this entity's zone information and another
        entity's zone information."
    ::= { zipRouter 5 }

zipInObsoletes OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ZIP Takedown or ZIP Bringup packets
        received by this entity. Note that as the ZIP
        Takedown and ZIP Bringup packets have been
        obsoleted, the receipt of one of these packets
        indicates that a node sent it in error."
    ::= { zipRouter 6 }
```



```
-- The zipRouterNetInfoTable is used to record information
-- about zipGetNetInfo and zipGetNetInfo Reply packets that
-- were received on each port for a router. This table
-- augments the atportTable.
```

```
zipRouterNetInfoTable OBJECT-TYPE
    SYNTAX SEQUENCE OF ZipRouterNetInfoEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The table of Net Info packets received by each port
         on this entity."
    ::= { zipRouter 7 }
```

```
zipRouterNetInfoEntry OBJECT-TYPE
    SYNTAX ZipRouterNetInfoEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The description of the Net Info packets received on
         a particular port on this entity. One such entry
         shall exist for each atport on this router entity.

         As an example, an instance of the zipInGetNetInfos
         object might be named zipInGetNetInfos.2"
    INDEX { atportIndex }
    ::= { zipRouterNetInfoTable 1 }
```

```
ZipRouterNetInfoEntry ::= SEQUENCE {
    zipInGetNetInfos      Counter,
    zipOutGetNetInfoReplies Counter,
    zipZoneOutInvalids    Counter,
    zipAddressInvalids     Counter
}
```

```
zipInGetNetInfos OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ZIP GetNetInfo packets received on
         this port by this entity."
    ::= { zipRouterNetInfoEntry 1 }
```

```
zipOutGetNetInfoReplies OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
```

## DESCRIPTION

"The number of ZIP GetNetInfo Reply packets sent out this port by this entity."

::= { zipRouterNetInfoEntry 2 }

## zipZoneOutInvalids OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"The number of times this entity has sent a ZIP GetNetInfo Reply with the zone invalid bit set in response to a GetNetInfo Request with an invalid zone name."

::= { zipRouterNetInfoEntry 3 }

## zipAddressInvalids OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

## DESCRIPTION

"The number of times this entity had to broadcast a ZIP GetNetInfo Reply because the GetNetInfo Request had an invalid address."

::= { zipRouterNetInfoEntry 4 }

-- The ZIP End Node Group

--

-- Implementation of this group is mandatory for all entities  
-- that implement ZIP

--

-- This group consists of ZIP variables that would be  
-- implemented by either a router or an end node.

-- The zipNetInfoTable is used to record information about  
-- zipGetNetInfo and zipGetNetInfo Reply packets that were  
-- received on each port of an entity. This table augments  
-- the atportTable.

## zipNetInfoTable OBJECT-TYPE

SYNTAX SEQUENCE OF ZipNetInfoEntry

ACCESS not-accessible

STATUS mandatory

## DESCRIPTION

"The table of Net Info packets received by each port on this entity."

::= { zipEndNode 1 }

## zipNetInfoEntry OBJECT-TYPE

SYNTAX ZipNetInfoEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"The description of the Net Info packets received on a particular port on this entity. One such entry shall exist for each atport on this entity.

As an example, an instance of the zipOutGetNetInfos object might be named zipOutGetNetInfos.2"

INDEX { atportIndex }

::= { zipNetInfoTable 1 }

ZipNetInfoEntry ::= SEQUENCE {

zipOutGetNetInfos Counter,

zipInGetNetInfoReplies Counter,

zipZoneInInvalids Counter

}

## zipOutGetNetInfos OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of ZIP GetNetInfo packets sent out this port by this entity."

::= { zipNetInfoEntry 1 }

## zipInGetNetInfoReplies OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of ZIP GetNetInfo Reply packets received on this port by this entity."

::= { zipNetInfoEntry 2 }

## zipZoneInInvalids OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of times this entity has received a ZIP GetNetInfo Reply with the zone invalid bit set because the corresponding GetNetInfo Request had an invalid zone name."

::= { zipNetInfoEntry 3 }

```

zipInErrors OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ZIP packets received by this entity
         that were rejected for any error."
    ::= { zipEndNode 2 }

-- The NBP Group
--
-- Implementation of this group is mandatory for all entities
-- that implement NBP

nbpTable OBJECT-TYPE
    SYNTAX SEQUENCE OF NbpEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The table of NBP services registered on this entity."
    ::= { nbp 1 }

nbpEntry OBJECT-TYPE
    SYNTAX NbpEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The description of an NBP service registered on this
         entity.

         As an example, an instance of the nbpZone object
         might be named nbpZone.2"
    INDEX { nbpIndex }
    ::= { nbpTable 1 }

NbpEntry ::= SEQUENCE {
    nbpIndex          INTEGER,
    nbpObject          ATName (SIZE (1..32)),
    nbpType            ATName (SIZE (1..32)),
    nbpZone            ATName,
    nbpState           INTEGER,
    nbpAddress         DdpSocketAddress,
    nbpEnumerator      INTEGER (0..255)
}

nbpIndex OBJECT-TYPE
    SYNTAX INTEGER

```

```
ACCESS read-write
STATUS mandatory
DESCRIPTION
    "The index of this NBP entry.  This index is unique
    with respect to the indexes of all other NBP entries,
    and shall remain constant throughout the lifetime
    of this object."
 ::= { nbpEntry 1 }

nbpObject OBJECT-TYPE
    SYNTAX ATName (SIZE (1..32))
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The name of the service described by this entity.
        When this variable is changed, the entity should
        perform an NBP registration using the new nbpObject."
    ::= { nbpEntry 2 }

nbpType OBJECT-TYPE
    SYNTAX ATName (SIZE (1..32))
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The type of the service described by this entity.
        When this variable is changed, the entity should
        perform an NBP registration using the new nbpType."
    ::= { nbpEntry 3 }

nbpZone OBJECT-TYPE
    SYNTAX ATName
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The zone the service described by this entity is
        registered in.  This must be the actual zone name,
        without any wildcard characters.  When this variable
        is changed, the entity should perform an NBP
        registration using the new nbpZone."
    ::= { nbpEntry 4 }

nbpState OBJECT-TYPE
    SYNTAX INTEGER {
        valid(1),
        registering(2), -- attempting to register the service
        registrationFailed(3),
        invalid(4)
    }
```

ACCESS read-write  
STATUS mandatory  
DESCRIPTION

"The state of this NBP entry.

When the registration for an entry in the nbpTable fails, it is an implementation-specific matter as to how long the entry will remain in the registrationFailed(3) state before moving to the invalid(4) state. Note that the entry may pass immediately from the registrationFailed state to the invalid state.

Setting this object to the value invalid(4) has the effect of invalidating the corresponding entry in the nbpTable. That is, it effectively disassociates 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 from agents tabular information corresponding to entries not currently in use. Proper interpretation of such entries requires examination of the relevant nbpState object."

::= { nbpEntry 5 }

nbpAddress OBJECT-TYPE  
SYNTAX DdpSocketAddress  
ACCESS read-write  
STATUS mandatory  
DESCRIPTION

"The DDP network, node, and socket number of this entity. If this is unspecified, for instance if the registration is on all ports of a multiport device, this object shall have the value of three octets of zero, followed by one octet of socket number."

::= { nbpEntry 6 }

nbpEnumerator OBJECT-TYPE  
SYNTAX INTEGER (0..255)  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The enumerator assigned to this entity."

::= { nbpEntry 7 }

nbpInLookUpRequests OBJECT-TYPE  
SYNTAX Counter

ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The number of NBP LookUp Requests received."  
 ::= { nbp 2 }

nbpInLookUpReplies OBJECT-TYPE  
    SYNTAX Counter  
    ACCESS read-only  
    STATUS mandatory  
    DESCRIPTION  
        "The number of NBP LookUp Replies received."  
 ::= { nbp 3 }

nbpInBroadcastRequests OBJECT-TYPE  
    SYNTAX Counter  
    ACCESS read-only  
    STATUS mandatory  
    DESCRIPTION  
        "The number of NBP Broadcast Requests received."  
 ::= { nbp 4 }

nbpInForwardRequests OBJECT-TYPE  
    SYNTAX Counter  
    ACCESS read-only  
    STATUS mandatory  
    DESCRIPTION  
        "The number of NBP Forward Requests received."  
 ::= { nbp 5 }

nbpOutLookUpReplies OBJECT-TYPE  
    SYNTAX Counter  
    ACCESS read-only  
    STATUS mandatory  
    DESCRIPTION  
        "The number of NBP LookUp Replies sent."  
 ::= { nbp 6 }

nbpRegistrationFailures OBJECT-TYPE  
    SYNTAX Counter  
    ACCESS read-only  
    STATUS mandatory  
    DESCRIPTION  
        "The number of times this node experienced a failure  
        in attempting to register an NBP entity."  
 ::= { nbp 7 }

```
nbpInErrors OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of NBP packets received by this entity
        that were rejected for any error."
    ::= { nbp 8 }

-- The ATEcho Group
--
-- Implementation of this group is mandatory for all
-- entities that implement ATEcho

atechoRequests OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of AppleTalk Echo requests received."
    ::= { atecho 1 }

atechoReplies OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of AppleTalk Echo replies sent."
    ::= { atecho 2 }

atechoOutRequests OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The count of AppleTalk Echo requests sent."
    ::= { atecho 3 }

atechoInReplies OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The count of AppleTalk Echo replies received."
    ::= { atecho 4 }
```



```
-- The ATP Group
--
-- Implementation of this group is mandatory for all entities
-- that implement ATP
```

```
atpInPkts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ATP packets received by this entity."
    ::= { atp 1 }
```

```
atpOutPkts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ATP packets sent by this entity."
    ::= { atp 2 }
```

```
atpTRequestRetransmissions OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times that a timeout occurred and a
        Transaction Request packet needed to be
        retransmitted by this host."
    ::= { atp 3 }
```

```
atpTResponseRetransmissions OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times a timeout was detected and a
        Transaction Response packet needed to be
        retransmitted by this host."
    ::= { atp 4 }
```

```
atpReleaseTimerExpiredCounts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times the release timer expired, as a
        result of which a Request Control Block had to be
```

```

        deleted."
 ::= { atp 5 }

```

atpRetryCountExceeded OBJECT-TYPE

```

    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION

```

"The number of times the retry count was exceeded,  
and an error was returned to the client of ATP."

```

 ::= { atp 6 }

```

atpListenerTable OBJECT-TYPE

```

    SYNTAX SEQUENCE OF AtpListenerEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION

```

"The atpListenerTable stores information for each ATP  
socket that has a listener."

```

 ::= { atp 7 }

```

atpListenerEntry OBJECT-TYPE

```

    SYNTAX AtpListenerEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION

```

"This atpListenerEntry contains information about a  
particular socket that has a socket listener."

As an example, an instance of the atpListenerStatus  
object might be named atpListenerStatus.0.80.220.3"

```

    INDEX { atpListenerAddress }
 ::= { atpListenerTable 1 }

```

```

AtpListenerEntry ::= SEQUENCE {
    atpListenerAddress  DdpSocketAddress,
    atpListenerStatus   INTEGER
}

```

atpListenerAddress OBJECT-TYPE

```

    SYNTAX DdpSocketAddress
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION

```

"The DDP address that this socket listener is bound  
to. If this socket listener isn't bound to a  
particular address, for instance if it is intended  
for all interfaces, this object shall have the value

of three octets of zero followed by one octet of  
socket number."  
 ::= { atpListenerEntry 1 }

atpListenerStatus OBJECT-TYPE

SYNTAX INTEGER {  
    valid(1),  
    invalid(2)  
}

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The status of this socket.

Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the atpListenerTable. That is, it effectively disassociates 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 from agents tabular information corresponding to entries not currently in use. Proper interpretation of such entries requires examination of the relevant atpListenerStatus object."

::= { atpListenerEntry 2 }

-- The PAP group

--

-- Implementation of this group is mandatory for all entities

-- that implement PAP

papInOpenConns OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of PAP Open Connection requests received  
by this entity."

::= { pap 1 }

papOutOpenConns OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of PAP Open Connection requests sent by  
this entity."  
 ::= { pap 2 }

papInDatas OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
"The number of PAP Data messages received by  
this entity."  
 ::= { pap 3 }

papOutDatas OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
"The number of PAP Data messages sent by  
this entity."  
 ::= { pap 4 }

papInCloseConns OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
"The number of PAP Close Connection requests  
received by this entity."  
 ::= { pap 5 }

papOutCloseConns OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
"The number of PAP Close Connection requests sent by  
this entity."  
 ::= { pap 6 }

papTickleTimeoutCloses OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
"The number of times the PAP entity on this node  
closed a connection because it didn't receive a  
Tickle message before its timer expired."

```
::= { pap 7 }
```

```
papServerTable OBJECT-TYPE
```

```
SYNTAX SEQUENCE OF PapServerEntry
```

```
ACCESS not-accessible
```

```
STATUS mandatory
```

```
DESCRIPTION
```

```
"A list of servers on this entity that are
accessible through the Printer Access Protocol."
```

```
::= { pap 8 }
```

```
papServerEntry OBJECT-TYPE
```

```
SYNTAX PapServerEntry
```

```
ACCESS not-accessible
```

```
STATUS mandatory
```

```
DESCRIPTION
```

```
"A set of information about a particular PAP server's
configuration and performance."
```

```
As an example, an instance of the papServerStatus
object might be named papServerStatus.1"
```

```
INDEX { papServerIndex }
```

```
::= { papServerTable 1 }
```

```
PapServerEntry ::= SEQUENCE {
```

```
    papServerIndex
```

```
    INTEGER,
```

```
    papServerListeningSocket
```

```
    DdpSocketAddress,
```

```
    papServerStatus
```

```
    DisplayString,
```

```
    papServerCompletedJobs
```

```
    Counter,
```

```
    papServerBusyJobs
```

```
    INTEGER,
```

```
    papServerFreeJobs
```

```
    INTEGER,
```

```
    papServerAuthenticationFailures
```

```
    Counter,
```

```
    papServerAccountingFailures
```

```
    Counter,
```

```
    papServerGeneralFailures
```

```
    Counter,
```

```
    papServerState
```

```
    INTEGER,
```

```
    papServerLastStatusMsg
```

```
    DisplayString
```

```
}
```

```
papServerIndex OBJECT-TYPE
```

```
SYNTAX INTEGER
```

```
ACCESS not-accessible
```

```
STATUS mandatory
```

```
DESCRIPTION
```

```
"An unique value for each Printer Access Protocol
Server."
```

```
::= { papServerEntry 1 }
```

papServerListeningSocket OBJECT-TYPE  
SYNTAX DdpSocketAddress  
ACCESS read-write  
STATUS mandatory  
DESCRIPTION  
    "The Server Listening Socket that this PAP server is  
    listening on."  
 ::= { papServerEntry 2 }

papServerStatus OBJECT-TYPE  
SYNTAX DisplayString  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The status string of this server. This is the  
    message as it would appear in a PAP Status Reply  
    from this server."  
 ::= { papServerEntry 3 }

papServerCompletedJobs OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The number of jobs that have been accepted and  
    successfully executed by this server."  
 ::= { papServerEntry 4 }

papServerBusyJobs OBJECT-TYPE  
SYNTAX INTEGER  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The number of GetNextJob calls that have accepted  
    and are currently executing a job."  
 ::= { papServerEntry 5 }

papServerFreeJobs OBJECT-TYPE  
SYNTAX INTEGER  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The minimum number of GetNextJob calls that are  
    currently waiting for a job."  
 ::= { papServerEntry 6 }

papServerAuthenticationFailures OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of times this PAP server rejected a job because the job was not correctly authenticated."

::= { papServerEntry 7 }

papServerAccountingFailures OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of times this PAP server rejected a job because the job did not fit some accounting rule, such as exceeding a quota."

::= { papServerEntry 8 }

papServerGeneralFailures OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of times this PAP server rejected a job for some reason other than authentication or accounting failures."

::= { papServerEntry 9 }

papServerState OBJECT-TYPE

SYNTAX INTEGER {

valid(1),

invalid(2)

}

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The state of this PAP Server entry.

Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the papServerTable. That is, it effectively disassociates 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 from agents tabular information corresponding to entries not currently

```
        in use.  Proper interpretation of such entries
        requires examination of the relevant papServerState
        object."
 ::= { papServerEntry 10 }

papServerLastStatusMsg OBJECT-TYPE
    SYNTAX DisplayString
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The last status message that was transmitted by
        this server."
 ::= { papServerEntry 11 }

-- The ASP Group
--
-- Implementation of this group is mandatory for all entities
-- that implement ASP

aspInputTransactions OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ASP requests and replies received by
        this entity.  Note that this is not necessarily the
        number of packets containing ASP transactions."
 ::= { asp 1 }

aspOutputTransactions OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ASP requests and replies sent by this
        entity.  Note that this is not necessarily the number
        of packets containing ASP transactions."
 ::= { asp 2 }

aspInOpenSessions OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ASP Open Session requests and replies
        received by this entity."
 ::= { asp 3 }
```



**aspOutOpenSessions OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of ASP Open Session requests and replies sent by this entity."

::= { asp 4 }

**aspInCloseSessions OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of ASP Close Session requests and replies received by this entity."

::= { asp 5 }

**aspOutCloseSessions OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of ASP Close Session requests and replies sent by this entity."

::= { asp 6 }

**aspNoMoreSessionsErrors OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of times an error condition was returned because this server implementation could not support another session."

::= { asp 7 }

**aspTickleTimeOutCloses OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of times the ASP entity on this node closed a connection because it didn't receive any messages from the remote end before its timer expired."

::= { asp 8 }

```

aspConnTable OBJECT-TYPE
    SYNTAX SEQUENCE OF AspConnEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A list of ASP connections on this entity."
    ::= { asp 9 }

aspConnEntry OBJECT-TYPE
    SYNTAX AspConnEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A set of information describing an ASP connection.

        As an example, an instance of the aspConnState object
        might be named
        aspConnState.0.80.220.135.0.80.239.119.12"
    INDEX { aspConnLocalAddress, aspConnRemoteAddress,
            aspConnID }
    ::= { aspConnTable 1 }

AspConnEntry ::= SEQUENCE {
    aspConnLocalAddress      DdpSocketAddress,
    aspConnRemoteAddress     DdpSocketAddress,
    aspConnID                INTEGER (1..255),
    aspConnLastReqNum        INTEGER (1..65535),
    aspConnServerEnd         INTEGER,
    aspConnState             INTEGER
}

aspConnLocalAddress OBJECT-TYPE
    SYNTAX DdpSocketAddress
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The local address of this ASP connection."
    ::= { aspConnEntry 1 }

aspConnRemoteAddress OBJECT-TYPE
    SYNTAX DdpSocketAddress
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The remote address of this ASP connection.  If
        this entry is in the listening mode, this object
        shall have a value of four octets of zero."
    ::= { aspConnEntry 2 }

```

aspConnID OBJECT-TYPE  
SYNTAX INTEGER (1..255)  
ACCESS not-accessible  
STATUS mandatory  
DESCRIPTION  
    "The remote Connection ID of this ASP connection. If  
    this entry is in the listening mode, this object  
    shall have a value of zero."  
 ::= { aspConnEntry 3 }

aspConnLastReqNum OBJECT-TYPE  
SYNTAX INTEGER (1..65535)  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The last request number on this ASP connection. If  
    this entry is in the listening mode, this object  
    shall have a value of zero."  
 ::= { aspConnEntry 4 }

aspConnServerEnd OBJECT-TYPE  
SYNTAX INTEGER {  
    sss(1), -- Server Session Socket  
    wss(2), -- Workstation Session Socket  
    sls(3) -- Server Listening Socket  
}  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "Specifies what mode the local session end is in."  
 ::= { aspConnEntry 5 }

aspConnState OBJECT-TYPE  
SYNTAX INTEGER {  
    open(1),  
    closed(2),  
    invalid(3)  
}  
ACCESS read-write  
STATUS mandatory  
DESCRIPTION  
    "The state of this ASP connection.  
    Setting this object to the value invalid(3) has the  
    effect of invalidating the corresponding entry in the  
    aspConnTable. That is, it effectively disassociates  
    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 from agents tabular information corresponding to entries not currently in use. Proper interpretation of such entries requires examination of the relevant aspConnState object."

```
::= { aspConnEntry 6 }
```

```
-- The ADSP Group
--
-- Implementation of this group is mandatory for all entities
-- that implement ADSP
```

```
adspInPkts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ADSP packets received by this entity."
    ::= { adsp 1 }
```

```
adspOutPkts OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of ADSP packets sent by this entity."
    ::= { adsp 2 }
```

```
adspInOctets OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of data octets contained in ADSP packets
        received by this entity. Note that this does not
        include EOM bits."
    ::= { adsp 3 }
```

```
adspOutOctets OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of data octets contained in ADSP packets
        sent by this entity. Note that this does not include
        EOM bits."
```

```
::= { adsp 4 }
```

adspInDataPkts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of ADSP data packets this entity has received."

```
::= { adsp 5 }
```

adspOutDataPkts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of ADSP data packets this entity has sent."

```
::= { adsp 6 }
```

adspTimeoutErrors OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of times the ADSP on this entity detected an expired connection timer."

```
::= { adsp 7 }
```

adspTimeoutCloseErrors OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of times the ADSP on this entity closed a connection because of too many timeouts."

```
::= { adsp 8 }
```

adspConnTable OBJECT-TYPE

SYNTAX SEQUENCE OF AdspConnEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"A list of ADSP connections on this entity."

```
::= { adsp 9 }
```

adspConnEntry OBJECT-TYPE

SYNTAX AdspConnEntry

```

ACCESS not-accessible
STATUS mandatory
DESCRIPTION
    "A set of information describing an ADSP connection.
    As an example, an instance of the adspConnState object
    might be named
    adspConnState.0.80.220.7.0.80.239.142.31231"
INDEX { adspConnLocalAddress, adspConnRemoteAddress,
        adspConnLocalConnID }
 ::= { adspConnTable 1 }

AdspConnEntry ::= SEQUENCE {
    adspConnLocalAddress      DdpSocketAddress,
    adspConnLocalConnID      INTEGER (0..65535),
    adspConnRemoteAddress     DdpSocketAddress,
    adspConnRemoteConnID     INTEGER (0..65535),
    adspConnState             INTEGER
}

adspConnLocalAddress OBJECT-TYPE
    SYNTAX DdpSocketAddress
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The local DDP address of this ADSP connection."
    ::= { adspConnEntry 1 }

adspConnLocalConnID OBJECT-TYPE
    SYNTAX INTEGER (0..65535)
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The local Connection ID of this ADSP connection.  If
        this entry specifies an ADSP listener, this value
        shall be zero."
    ::= { adspConnEntry 2 }

adspConnRemoteAddress OBJECT-TYPE
    SYNTAX DdpSocketAddress
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The remote DDP address of this ADSP connection.  If
        this entry specifies an ADSP listener, this value
        shall be zero."
    ::= { adspConnEntry 3 }

adspConnRemoteConnID OBJECT-TYPE

```

SYNTAX INTEGER (0..65535)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The remote Connection ID of this ADSP connection.

If this entry specifies an ADSP listener, this value shall be zero."

::= { adspConnEntry 4 }

adspConnState OBJECT-TYPE

SYNTAX INTEGER {

open(1),

localHalfOpen(2),

remoteHalfOpen(3),

listening(4),

closed(5),

invalid(6)

}

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The state of this ADSP connection. The state is open if both ends are established. If only one end is established, then the state is half-open. If neither end is established, then the state is closed. If an ADSP server is listening on a socket and is not yet connected, its state is set to listening, and the adspConnRemoteAddress, adspConnRemoteSocket, adspConnRemoteConnID, and adspConnRemoteWindowSize are all set to zero.

Setting this object to the value invalid(6) has the effect of invalidating the corresponding entry in the adspConnTable. That is, it effectively disassociates 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 from agents tabular information corresponding to entries not currently in use. Proper interpretation of such entries requires examination of the relevant adspConnState object."

::= { adspConnEntry 5 }

```
-- The ATPortPtoP Group
--
-- Implementation of this group is mandatory for all entities
-- that implement AppleTalk point-to-point links
```

```
atportPtoPTable OBJECT-TYPE
    SYNTAX SEQUENCE OF AtportPtoPEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A list of AppleTalk point-to-point connections for
        this entity."
    ::= { atportptop 1 }
```

```
atportPtoPEntry OBJECT-TYPE
    SYNTAX AtportPtoPEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The description of one of the AppleTalk
        point-to-point connections on this entity.

        As an example, an instance of the
        atportPtoPRemoteAddress object might be named
        atportPtoPRemoteAddress.2"
    INDEX { atportPtoPIndex }
    ::= { atportPtoPTable 1 }
```

```
AtportPtoPEntry ::= SEQUENCE {
    atportPtoPIndex          INTEGER,
    atportPtoPProtocol       OBJECT IDENTIFIER,
    atportPtoPRemoteName     DisplayString,
    atportPtoPRemoteAddress  OCTET STRING,
    atportPtoPPortIndex      INTEGER,
    atportPtoPStatus         INTEGER
}
```

```
atportPtoPIndex OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A unique value for each AppleTalk point-to-point
        connection. Its value is between 1 and the total
        number of AppleTalk point-to-point connections. The
        value for each connection must remain constant at
        least from the re-initialization of the entity's
        network management system to the next
```



```
        re-initialization."
 ::= { atportPtoPEntry 1 }

atportPtoPProtocol OBJECT-TYPE
    SYNTAX OBJECT IDENTIFIER
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The protocol type used over the point-to-point
        connection."
 ::= { atportPtoPEntry 2 }

atportPtoPRemoteName OBJECT-TYPE
    SYNTAX DisplayString
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "A text string containing the network node name of the
        entity at the other end of the point-to-point link.
        If the name is unknown or undefined, then this
        string is zero length."
 ::= { atportPtoPEntry 3 }

atportPtoPRemoteAddress OBJECT-TYPE
    SYNTAX OCTET STRING
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The network address of the entity at the other end
        of the point-to-point link in network byte order.
        The format of this address can be determined
        by examining the atportType corresponding to this
        entry. If the address is unknown or undefined, then
        this string is zero length."
 ::= { atportPtoPEntry 4 }

atportPtoPPortIndex OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
        "The AppleTalk port associated with this
        point-to-point connection. The interface identified
        by a particular value of this index is the same
        interface as identified by the same value of
        atportIndex."
 ::= { atportPtoPEntry 5 }
```

## atportPtoPStatus OBJECT-TYPE

```
SYNTAX INTEGER {
    valid(1),
    invalid(2)
}
```

```
ACCESS read-write
```

```
STATUS mandatory
```

## DESCRIPTION

"The status of this entry in the atportPtoPTable.

Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the atportPtoPTable. That is, it effectively disassociates 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 from agents tabular information corresponding to entries not currently in use. Proper interpretation of such entries requires examination of the relevant atportPtoPStatus object."

```
::= { atportPtoPEntry 6 }
```

```
atportPtoPProtoOids OBJECT IDENTIFIER ::= { atportptop 2 }
```

```
-- A list of values to be used for the atportPtoPProtocol
-- variable.
```

```
-- When new protocols are defined, their oids may be defined
-- in separate MIB documents in different branches of the tree.
```

```
pToPProtoOther OBJECT IDENTIFIER ::= { atportPtoPProtoOids 1 }
```

```
pToPProtoAurp OBJECT IDENTIFIER ::= { atportPtoPProtoOids 2 }
```

```
pToPProtoCaymanUdp OBJECT IDENTIFIER ::=
```

```
    { atportPtoPProtoOids 3 }
```

```
pToPProtoAtkvmsDecnetIV OBJECT IDENTIFIER ::=
```

```
    { atportPtoPProtoOids 4 }
```

```
pToPProtoLiaisonUdp OBJECT IDENTIFIER ::=
```

```
    { atportPtoPProtoOids 5 }
```

```
pToPProtoIpx OBJECT IDENTIFIER ::= { atportPtoPProtoOids 6 }
```

```
pToPProtoShivaIp OBJECT IDENTIFIER ::=
```

```
    { atportPtoPProtoOids 7 }
```

```

-- The Per Port Counters Group
--
-- Implementation of this group is optional.

perPortTable OBJECT-TYPE
    SYNTAX SEQUENCE OF PerPortEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The table of per-port statistics for this entity."
    ::= { perPort 1 }

perPortEntry OBJECT-TYPE
    SYNTAX PerPortEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "The statistics available for a particular port on
        this entity.

        As an example, an instance of the perPortAarpInProbes
        object might be named perPortAarpInProbes.2"
    INDEX { atportIndex }
    ::= { perPortTable 1 }

PerPortEntry ::= SEQUENCE {
    perPortAarpInProbes          Counter,
    perPortAarpOutProbes         Counter,
    perPortAarpInReqs            Counter,
    perPortAarpOutReqs           Counter,
    perPortAarpInRsps            Counter,
    perPortAarpOutRsps           Counter,
    perPortDdpInReceives         Counter,
    perPortDdpInLocalDatagrams   Counter,
    perPortDdpNoProtocolHandlers Counter,
    perPortDdpTooShortErrors      Counter,
    perPortDdpTooLongErrors       Counter,
    perPortDdpChecksumErrors      Counter,
    perPortDdpForwRequests        Counter,
    perPortRtmpInDataPkts         Counter,
    perPortRtmpOutDataPkts        Counter,
    perPortRtmpInRequestPkts      Counter,
    perPortRtmpRouteDeletes       Counter,
    perPortZipInZipQueries        Counter,
    perPortZipInZipReplies        Counter,
    perPortZipInZipExtendedReplies Counter,
    perPortZipZoneConflictErrors  Counter,
    perPortZipInErrors            Counter,

```

```
perPortNbpInLookUpRequests      Counter,
perPortNbpInLookUpReplies       Counter,
perPortNbpInBroadcastRequests   Counter,
perPortNbpInForwardRequests     Counter,
perPortNbpOutLookUpReplies      Counter,
perPortNbpRegistrationFailures  Counter,
perPortNbpInErrors              Counter,
perPortEchoRequests             Counter,
perPortEchoReplies              Counter
}

perPortAarpInProbes OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of AARP Probe packets received
         by this entity on this port."
    ::= { perPortEntry 1 }

perPortAarpOutProbes OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of AARP Probe packets sent by
         this entity on this port."
    ::= { perPortEntry 2 }

perPortAarpInReqs OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of AARP Request packets received
         by this entity on this port."
    ::= { perPortEntry 3 }

perPortAarpOutReqs OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of AARP Request packets sent by
         this entity on this port."
    ::= { perPortEntry 4 }
```

```
perPortAarpInRsps OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of AARP Response packets received
        by this entity on this port."
    ::= { perPortEntry 5 }

perPortAarpOutRsps OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of AARP Response packets sent by
        this entity on this port."
    ::= { perPortEntry 6 }

perPortDdpInReceives OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of input datagrams received by DDP
        on this port, including those received in error."
    ::= { perPortEntry 7 }

perPortDdpInLocalDatagrams OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of input DDP datagrams on this
        port for which this entity was their final DDP
        destination."
    ::= { perPortEntry 8 }

perPortDdpNoProtocolHandlers OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The total number of DDP datagrams addressed to this
        entity on this port that were addressed to an upper
        layer protocol for which no protocol handler
        existed."
    ::= { perPortEntry 9 }
```

**perPortDdpTooShortErrors OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The total number of input DDP datagrams on this port dropped because the received data length was less than the data length specified in the DDP header or the received data length was less than the length of the expected DDP header."

::= { perPortEntry 10 }

**perPortDdpTooLongErrors OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The total number of input DDP datagrams on this port dropped because they exceeded the maximum DDP datagram size."

::= { perPortEntry 11 }

**perPortDdpChecksumErrors OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The total number of input DDP datagrams on this port for which this DDP entity was their final destination, and which were dropped because of a checksum error." ::= { perPortEntry 12 }

**perPortDdpForwRequests OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of input datagrams on this port for which this entity was not their final DDP destination, as a result of which an attempt was made to find a route to forward them to that final destination."

::= { perPortEntry 13 }

**perPortRtmpInDataPkts OBJECT-TYPE**

SYNTAX Counter

ACCESS read-only

STATUS mandatory  
DESCRIPTION  
    "A count of the number of good RTMP data packets  
    received by this entity on this port."  
 ::= { perPortEntry 14 }

perPortRtmpOutDataPkts OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "A count of the number of RTMP packets sent by this  
    entity on this port."  
 ::= { perPortEntry 15 }

perPortRtmpInRequestPkts OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "A count of the number of good RTMP Request packets  
    received by this entity on this port."  
 ::= { perPortEntry 16 }

perPortRtmpRouteDeletes OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "A count of the number of times RTMP deletes a route  
    on this port because it was aged out of the table."  
 ::= { perPortEntry 17 }

perPortZipInZipQueries OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION  
    "The number of ZIP Queries received by this entity  
    on this port."  
 ::= { perPortEntry 18 }

perPortZipInZipReplies OBJECT-TYPE  
SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

"The number of ZIP Replies received by this entity  
        on this port."  
 ::= { perPortEntry 19 }

perPortZipInZipExtendedReplies OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

        "The number of ZIP Extended Replies received by this  
        entity on this port."  
 ::= { perPortEntry 20 }

perPortZipZoneConflictErrors OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

        "The number of times a conflict has been detected on  
        this port between this entity's zone information and  
        another entity's zone information."  
 ::= { perPortEntry 21 }

perPortZipInErrors OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

        "The number of ZIP packets received by this entity  
        on this port that were rejected for any error."  
 ::= { perPortEntry 22 }

perPortNbpInLookUpRequests OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

        "The number of NBP LookUp Requests received on this  
        port."  
 ::= { perPortEntry 23 }

perPortNbpInLookUpReplies OBJECT-TYPE

SYNTAX Counter  
ACCESS read-only  
STATUS mandatory  
DESCRIPTION

        "The number of NBP LookUp Replies received on this



```
        port."
 ::= { perPortEntry 24 }

perPortNbpInBroadcastRequests OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of NBP Broadcast Requests received on
        this port."
 ::= { perPortEntry 25 }

perPortNbpInForwardRequests OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of NBP Forward Requests received on this
        port."
 ::= { perPortEntry 26 }

perPortNbpOutLookUpReplies OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of NBP LookUp Replies sent on this port."
 ::= { perPortEntry 27 }

perPortNbpRegistrationFailures OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of times this node experienced a failure
        in attempting to register an NBP entity on this
        port."
 ::= { perPortEntry 28 }

perPortNbpInErrors OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of NBP packets received by this entity
        on this port that were rejected for any error."
 ::= { perPortEntry 29 }
```

```
perPortEchoRequests OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The number of AppleTalk Echo requests received on
         this port."
    ::= { perPortEntry 30 }

perPortEchoReplies OBJECT-TYPE
    SYNTAX Counter
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
        "The count of AppleTalk Echo replies received on
         this port."
    ::= { perPortEntry 31 }

END
```

## 6. Acknowledgments

This document was produced by the IETF AppleTalk-IP Working Group.

In addition, the contribution of the following individuals is also acknowledged:

```
Greg Bruell, Wellfleet
Phil Budne, Shiva
Robert Jeckell, 3Com
Greg Merrell, DEC
Greg Minshall, Novell, Inc.
Bob Morgan, Stanford University
Brad Parker, FCR
Marshall T. Rose, Dover Beach Consulting
Wayne Tackabury, Cayman
Jonathan Wenocur, Shiva
```

## 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, IAB, August 1989.
- [3] Rose M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based internets", STD 16, RFC 1155, Performance Systems International, Hughes LAN Systems, May 1990.
- [4] McCloghrie K., and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets", RFC 1156, Hughes LAN Systems, Performance Systems International, May 1990.
- [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] Rose, M., Editor, "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] Rose, M., and K. McCloghrie, Editors, "Concise MIB Definitions", STD 16, RFC 1212, Performance Systems International, Hughes LAN Systems, March 1991.
- [10] Gursharan S., Andrews, R., and A. Oppenheimer, "Inside AppleTalk", Second Edition, Addison Wesley, 1990.

## Security Considerations

Security issues are not discussed in this memo.

## 9. Authors' Addresses

Steven Waldbusser  
Carnegie Mellon University  
5000 Forbes Ave.  
Pittsburgh, PA 15213

Phone: 412-268-6628  
EMail: waldbusser@cmu.edu

Karen Frisa  
FORE Systems, Inc.  
174 Thorn Hill Road  
Warrendale, PA 15086-7535

Phone: 412-772-6541  
EMail: kfrisa@fore.com

