

## Contents

Contents .....	1
Introduction.....	2
Management Information Base (MIB).....	3
MIB Hierarchy .....	4
MIB Structure .....	5
Object Identifier .....	5
Tables.....	5
Private Extensions.....	5
Criteria and Philosophy for standardised MIB .....	5
Naming an Object .....	6
Object identifiers.....	6
Object and Object identifiers .....	6
MIB-II.....	7
System Group.....	7
Interfaces Group.....	9
Address Translation Group .....	16
IP Group.....	17
ICMP Group.....	24
TCP Group .....	25
EGP Group.....	29
SNMP Group .....	31

## Introduction

In order to operate, each device on a network (such as a switch, a router or a workstation) needs to keep a store of operational information which it uses to perform its task. In the example of a router this information would include routing tables, interface information etc.

A Network Management System (NMS) needs to be able to access this information in order to ensure that the devices are operating in the required manner. Each device will store this information internally in its own bespoke format and so, unaided, it would be complex for the NMS to understand all the formats used by the different manufacturers of the different devices on the network.

Consequently, a standardised presentation format for this information has been defined so that, no matter what the internal format of the device's data, it is presented by the device to the NMS in a standard way. This saves the NMS from having to be familiar with many proprietary formats. It is the responsibility of software within the device to translate between the internal and the standard formats. This software is called *the agent*.

This virtual database on the device can be thought of as an information warehouse and is called the Management Information Base (MIB).

This document will discuss the idea of the MIB in more detail and, in particular, the subset known as MIB-II, which is the minimal subset that all network devices must support if they are to claim to provide a Simple Network Management Protocol (SNMP) interface.

## Management Information Base (MIB)

Network management takes place between two major types of systems: those in control, called *managing systems*, and those observed and controlled, called *managed systems*. The most common managing system is the Network Management System (NMS). Managed systems can include hosts, servers, or network components such as routers or intelligent repeaters.

To promote interoperability, cooperating systems must adhere to a common framework and a common language, called a *protocol*. In the Internet network management framework, that protocol is the Simple Network Management Protocol (SNMP).

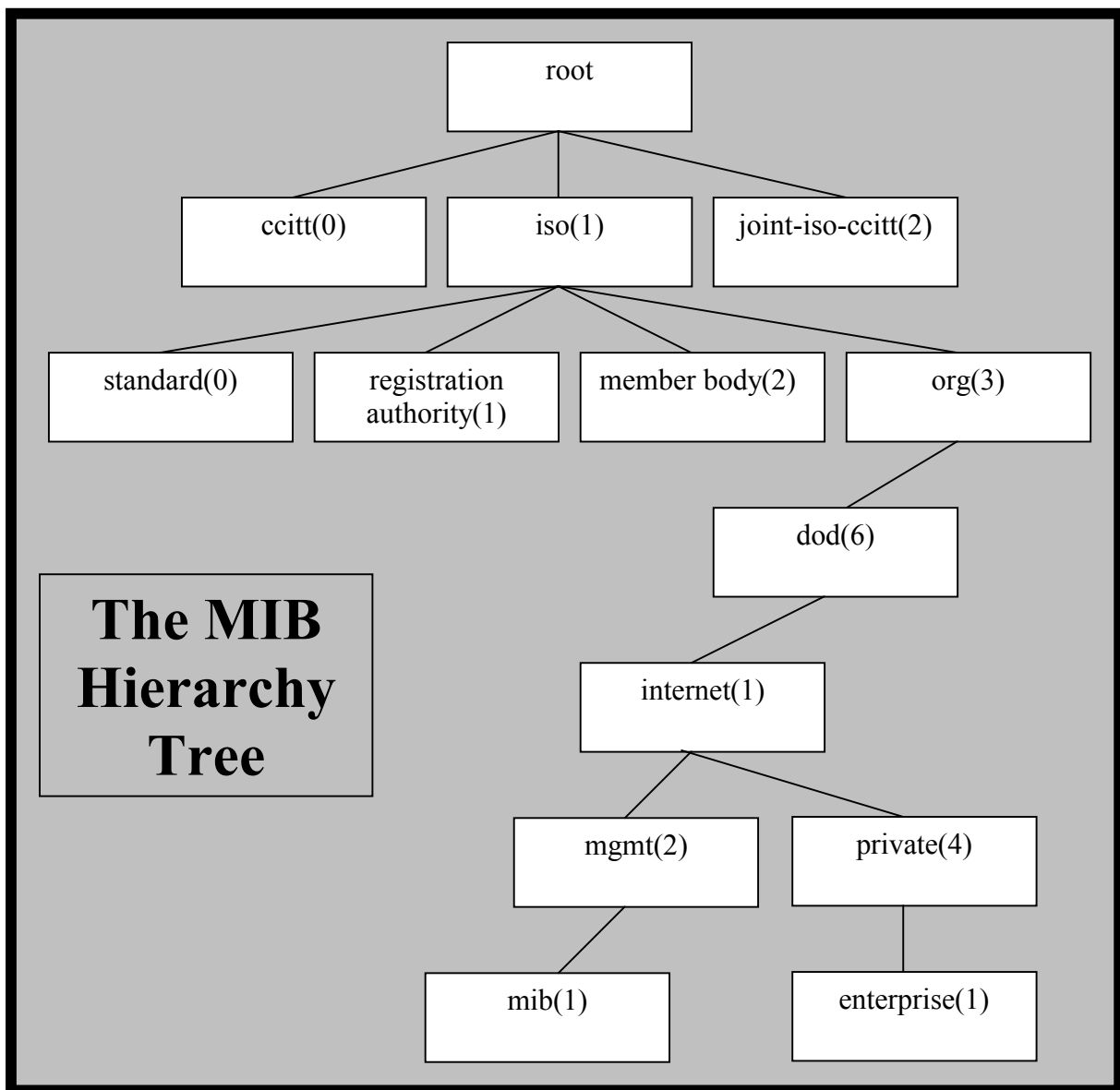
In a managed device, specialised low-impact software modules, called *agents*, access information about the device and make it available to the NMS. Managed devices maintain values for a number of variables and report those, as required, to the NMS. For example, an agent might report such data as the number of bytes and packets in and out of the device, or the number of broadcast messages sent and received. In the Internet network management framework each variable is referred to as a *managed object*, which is anything that an agent can access and report back to the NMS.

All managed objects are contained in the MIB, which is a database of the managed objects. The managed objects, or variables, can be set or read to provide information on network devices and interfaces. An NMS can control a managed device by sending a message to an agent of that managed device requiring the device to change the value of one or more of its variables.

## MIB Hierarchy

The MIB structure is logically represented by a tree hierarchy. The root of the tree is unnamed and splits into three main branches: Consultative Committee for International Telegraph and Telephone (CCITT), International Organization for Standardization (ISO), and joint ISO/CCITT.

These branches and those that fall below each category have short text strings and integers to identify them. Text strings describe *object names*, while integers allow computer software to create compact, encoded representations of the names. For example, the Internet standard MIB-II is represented by the object identifier *1.3.6.1.2.1*. It can also be expressed as *iso.org.dod.internet.mgmt.mib*.



The relevant part of the top of the tree is shown above.

# MIB Structure

## ***Object Identifier***

Each MIB variable is assigned an object identifier (OID). The *object identifier* is the sequence of numeric labels on the nodes along a path from the root to the object. For example, the MIB variable *sysDescr* in the system group of MIB-II is indicated by the number 1. The object identifier for *sysDescr* is therefore *1.3.6.1.2.1.1.1* or *iso.org.dod.internet.mgmt.mib.system.sysDescr*.

## ***Tables***

When network management protocols use names of MIB variables in messages, each name has an appended suffix. This suffix is called an *instance identifier*.

For simple variables, the instance identifier *0* refers to the instance of the variable with that name. A MIB can also contain tables of related variables. In this instance, the instance identifier refers to the table row (counted from 1).

## ***Private Extensions***

MIB-II contains the data that is deemed to be common between all network devices. However, different network devices may have different additional data that must be managed to ensure smooth operation. Each manufacturer will create its own MIB extensions for its own equipment and these extensions will be located in the *enterprises* part of the MIB hierarchy.

For example, the Case Communications equipment-specific extensions will be located under *1.3.6.1.4.1.144*.

## ***Criteria and Philosophy for standardised MIB***

- Objects have to be uniquely named.
- Objects have to be essential.
- Abstract structure of the MIB needs to be universal.
- For the standard MIB, maintain only a small number of objects.
- Allow for private extensions.
- Object must be general and not too device dependant.
- Objects cannot be easily derivable from their objects.
- If agent is to be SNMP manageable then it is mandatory to implement the Internet MIB (currently given as MIB-II in RFC 1213).

## ***Naming an Object***

- Universal unambiguous identification of arbitrary objects.
- Can be achieved by using a hierarchical tree.
- Based on the Object Identification Scheme defined by OSI.

## ***Object identifiers***

- Object name is given by its name in the tree.
- All child nodes are given unique integer values within that new sub-tree.
- Children can be parents of further child sub-tree (i.e. they have subordinates) where the numbering scheme is recursively applied.
- The Object Identifier (or name) of an object is the sequence of non-negative Integer values traversing the tree to the node required.
- Allocation of an integer value for a node in the tree is an act of registration by whoever has delegated authority for that sub tree.
- This process can go to an arbitrary depth.
- If a node has children then it is an aggregate node.
- Children of the same parent cannot have the same integer value.

## ***Object and Object identifiers***

- Object is named or identified by the sequence of integers in traversing the tree to the object type required.
- This does not identify an instance of the object.
- The Object Identifier (OID) is shown in a few ways with a.b.c.d.e being the preferred.
- For SNMP it is the Internet sub-tree for constructing OIDs for SNMP manageable agents.

## MIB-II

MIB-II (RFC 1213) is the current standard definition of the common virtual file store for SNMP manageable objects.

It has 10 basic groups:

Name	OID
system	mib.1
interfaces	mib.2
at	mib.3
ip	mib.4
icmp	mib.5
tcp	mib.6
udp	mib.7
egp	mib.8
transmission	mib.10
snmp	mib.11

If an agent implements any group then it has to implement all of the managed objects within that group.

An agent does not have to implement all groups.

Note: MIB-I (which was the precursor to MIB-II and which has been deprecated with the arrival of MIB-II) and MIB-II have same OID (i.e. position in the internet subtree).

The following tables describe the managed objects that reside within these groups.

### **System Group**

This is a collection of objects common to all managed systems.

Its OID is *mib.1*.

Object Name	SubID	Access	Description
sysDescr	1	read-only	A textual description of the entity. This value should include the full name and version identification of the system's hardware type, software operating-system, and networking software.
sysObjectID	2	read-only	The vendor's authoritative identification of the network management subsystem contained in the entity. This value is

			allocated within the SMI enterprises subtree (1.3.6.1.4.1) and provides an easy and unambiguous means for determining 'what kind of box' is being managed. For example, if vendor 'Flintstones, Inc.' was assigned the subtree 1.3.6.1.4.1.4242, it could assign the identifier 1.3.6.1.4.1.4242.1.1 to its 'Fred Router'.												
sysUpTime	3	read-only	The time (in hundredths of a second) since the network management portion of the system was last re-initialised.												
sysContact	4	read-write	The textual identification of the contact person for this managed node, together with information on how to contact this person. If no contact information is known, the value is the zero-length string.												
sysName	5	read-write	An administratively-assigned name for this managed node. By convention, this is the node's fully-qualified domain name. If the name is unknown, the value is the zero-length string.												
sysLocation	6	read-write	The physical location of this node (e.g. 'telephone closet, 3rd floor'). If the location is unknown, the value is the zero-length string.												
sysServices	7	read-only	<p>A value which indicates the set of services that this entity may potentially offers. The value is a sum. This sum initially takes the value zero, Then, for each layer, L, in the range 1 through 7, that this node performs transactions for, <math>2^{L-1}</math> is added to the sum.</p> <p>For example, a node which performs only routing functions would have a value of 4 (<math>2^{3-1}</math>). In contrast, a node which is a host offering application services would have a value of 72 (<math>2^{4-1} + 2^{7-1}</math>). Note that in the context of the Internet suite of protocols, values should be calculated accordingly:</p> <table border="1"> <thead> <tr> <th>Layer</th> <th>Functionality</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>physical (e.g. repeaters)</td> </tr> <tr> <td>2</td> <td>datalink/subnetwork (e.g. bridges)</td> </tr> <tr> <td>3</td> <td>internet (e.g. supports IP)</td> </tr> <tr> <td>4</td> <td>end-to-end (e.g. supports TCP)</td> </tr> <tr> <td>7</td> <td>applications (e.g. supports SMTP)</td> </tr> </tbody> </table> <p>For systems including OSI protocols, layers 5 and 6 may also be counted.</p>	Layer	Functionality	1	physical (e.g. repeaters)	2	datalink/subnetwork (e.g. bridges)	3	internet (e.g. supports IP)	4	end-to-end (e.g. supports TCP)	7	applications (e.g. supports SMTP)
Layer	Functionality														
1	physical (e.g. repeaters)														
2	datalink/subnetwork (e.g. bridges)														
3	internet (e.g. supports IP)														
4	end-to-end (e.g. supports TCP)														
7	applications (e.g. supports SMTP)														
sysORLastChange	8	read-only	The value of sysUpTime at the time of the most recent change in state or value of any instance of sysORID.												

sysORTable	9	not-accessible	The (conceptual) table listing the capabilities of the local SNMPv2 entity acting in an agent role with respect to various MIB modules. SNMPv2 entities having dynamically- configurable support of MIB modules will have a dynamically-varying number of conceptual rows.  This table is expanded below.
------------	---	----------------	---

The *sysOrTable* is a collection of objects which describe the SNMPv2 entity's (statically and dynamically configurable) support of various MIB modules.

Object Name	SubID	Access	Description
sysOREntry	9	not-accessible	An entry (conceptual row) in the sysORTable.

sysORIndex	9.1.1	not-accessible	The auxiliary variable used for identifying instances of the columnar objects in the sysORTable.
sysORID	9.1.2	read-only	An authoritative identification of a capabilities statement with respect to various MIB modules supported by the local SNMPv2 entity acting in an agent role.
sysORDescr	9.1.3	read-only	A textual description of the capabilities identified by the corresponding instance of sysORID.
sysORUpTime	9.1.4	read-only	The value of sysUpTime at the time this conceptual row was last instantiated.

### ***Interfaces Group***

This is a collection of information about the device's interfaces.

Its OID is *mib.2*.

Object Name	SubID	Access	Description
ifNumber	1	read-only	The number of network interfaces (regardless of their current state) present on this system.
ifTable	2	not-accessible	A list of interface entries. The number of entries is given by the value of ifNumber.

The *ifTable* contains information on the entity's interfaces. Each sub-layer below the internetwork-layer of a network interface is considered to be an interface.

Object Name	SubID	Access	Description
ifEntry	2	not-accessible	An entry containing management information applicable to a particular interface.

ifIndex	2.1.1	read-only	A unique value, greater than zero, for each interface. It is recommended that values are assigned contiguously starting from 1. The value for each interface sub-layer must remain constant at least from one re-initialisation of the entity's network management system to the next re-initialisation.
ifDescr	2.1.2	read-only	A textual string containing information about the interface. This string should include the name of the manufacturer, the product name and the version of the interface hardware/software.
ifType	2.1.3	read-only	The type of interface. Additional values for ifType are assigned by the Internet Assigned Numbers Authority (IANA), through updating the syntax of the IANAifType textual convention.  The table below shows the values defined for this object.
ifMtu	2.1.4	read-only	The size of the largest packet which can be sent/received on the interface, specified in octets. For interfaces that are used for transmitting network datagrams, this is the size of the largest network datagram that can be sent on the interface.
ifSpeed	2.1.5	read-only	An estimate of the interface's current bandwidth in bits per second. For interfaces which do not vary in bandwidth or for those where no accurate estimation can be made, this object should contain the nominal bandwidth. If the bandwidth of the interface is greater than the maximum value reportable by this object then this object should report its maximum value (4,294,967,295) and ifHighSpeed must be used to report the interface's speed. For a sub-layer which has no concept of bandwidth, this object should be zero.
ifPhysAddress	2.1.6	read-only	The interface's address at its protocol sub-layer. The interface's media-specific MIB must define the bit and byte ordering and format of the value contained by this object. For interfaces which do not have

			such an address (e.g. a serial line), this object should contain an octet string of zero length.
ifAdminStatus	2.1.7	read-write	The desired state of the interface. The testing(3) state indicates that no operational packets can be passed. When a managed system initialises, all interfaces start with ifAdminStatus in the down(2) state. As a result of either explicit management action or per configuration information retained by the managed system, ifAdminStatus is then changed to either the up(1) or testing(3) states (or remains in the down(2) state).
ifOperStatus	2.1.8	read-only	The current operational state of the interface. The testing(3) state indicates that no operational packets can be passed. If ifAdminStatus is down(2) then ifOperStatus should be down(2). If ifAdminStatus is changed to up(1) then ifOperStatus should change to up(1) if the interface is ready to transmit and receive network traffic; it should change to dormant(5) if the interface is waiting for external actions (such as a serial line waiting for an incoming connection); it should remain in the down(2) state if and only if there is a fault that prevents if from going to the up(1) state.
ifLastChange	2.1.9	read-only	The value of sysUpTime at the time the interface entered its current operational state. If the current state was entered prior to the last re-initialisation of the local network management subsystem, then this object contains a zero value.
ifInOctets	2.1.10	read-only	The total number of octets received on the interface, including framing characters.
ifInUcastPkts	2.1.11	read-only	The number of packets, delivered by this sub-layer to a higher (sub-)layer, which were not addressed to a multicast or broadcast address at this sub-layer.
ifInNUcastPkts	2.1.12	read-only	The number of packets, delivered by this sub-layer to a higher (sub-)layer, which were addressed to a multicast or broadcast address at this sub-layer. This object is deprecated in favour of ifInMulticastPkts and ifInBroadcastPkts.
ifInDiscards	2.1.13	read-only	The number of inbound packets which were chosen to be discarded even though no errors had been detected to prevent their being deliverable to a higher-layer protocol. One possible reason for discarding such a packet could be to free up buffer space.
ifInErrors	2.1.14	read-only	For packet-oriented interfaces, the number

			of inbound packets that contained errors preventing them from being deliverable to a higher-layer protocol. For character-oriented or fixed-length interfaces, the number of inbound transmission units that contained errors preventing them from being deliverable to a higher-layer protocol.
ifInUnknownProtos	2.1.15	read-only	For packet-oriented interfaces, the number of packets received via the interface which were discarded because of an unknown or unsupported protocol. For character-oriented or fixed-length interfaces which support protocol multiplexing the number of transmission units received via the interface which were discarded because of an unknown or unsupported protocol. For any interface which does not support protocol multiplexing, this counter will always be 0.
ifOutOctets	2.1.16	read-only	The total number of octets transmitted out of the interface, including framing characters.
ifOutUcastPkts	2.1.17	read-only	The total number of packets that higher-level protocols requested be transmitted, and which were not addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent.
ifOutNUcastPkts	2.1.18	read-only	The total number of packets that higher-level protocols requested be transmitted, and which were addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent. This object is deprecated in favour of ifOutMulticastPkts and ifOutBroadcastPkts.
ifOutDiscards	2.1.19	read-only	The number of outbound packets which were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space.
ifOutErrors	2.1.20	read-only	For packet-oriented interfaces, the number of outbound packets that could not be transmitted because of errors. For character-oriented or fixed-length interfaces, the number of outbound transmission units that could not be transmitted because of errors.
ifOutQLen	2.1.21	read-only	The length of the output packet queue (in packets).
ifSpecific	2.1.22	read-only	A reference to MIB definitions specific to the particular media being used to realise the interface. It is recommended that this

			value point to an instance of a MIB object in the media-specific MIB, i.e. that this object have the semantics associated with the InstancePointer textual convention defined in RFC 1443. In fact, it is recommended that the media-specific MIB specify what value ifSpecific should/can take for values of ifType. If no MIB definitions specific to the particular media are available, the value should be set to the OBJECT IDENTIFIER { 0 0 }.
--	--	--	---

*ifType* values are published periodically by the Internet Assigned Numbers Authority (IANA), in either the Assigned Numbers RFC, or some derivative of it specific to Internet Network Management number assignments. The latest arrangements can be obtained by contacting the IANA.

The current values are:

Value	Description
other(1)	none of the following
regular1822(2)	
hdh1822(3)	
ddnX25(4)	
rfc877x25(5)	
ethernetCsmacd(6)	
iso88023Csmacd(7)	
iso88024TokenBus(8)	
iso88025TokenRing(9)	
iso88026Man(10)	
starLan(11)	
proteon10Mbit(12)	
proteon80Mbit(13)	
hyperchannel(14)	
fdi(15)	
lapb(16)	
sdlc(17)	
ds1(18)	DS1-MIB
e1(19)	Obsolete see DS1 MIB
basicISDN(20)	
primaryISDN(21)	
propPointToPointSerial(22)	proprietary serial
ppp(23)	
softwareLoopback(24)	
eon(25)	CLNP over IP
ethernet3Mbit(26)	
nsip(27)	XNS over IP
slip(28)	generic SLIP
ultra(29)	ULTRA technologies
ds3(30)	DS3 MIB
sip(31)	SMDS, coffee
frameRelay(32)	DTE only.

rs232(33)	
para(34)	Parallel port
arcnet(35)	arcnet
arcnetPlus(36)	arcnet plus
atm(37)	ATM cells
miox25(38)	
sonet(39)	SONET or SDH
x25ple(40)	
iso88022llc(41)	
localTalk(42)	
smDsDxi(43)	
frameRelayService(44)	FRNETSERV MIB
v35(45)	
hssi(46)	
hippi(47)	
modem(48)	Generic modem
aal5(49)	AAL5 over ATM
sonetPath(50)	
sonetVT(51)	
smDslcip(52)	SMDS InterCarrier Interface
propVirtual(53)	proprietary virtual/internal
propMultiplexor(54)	proprietary multiplexing
ieee80212(55)	100BaseVG
fibreChannel(56)	Fibre Channel
hippiInterface(57)	HIPPI interfaces
frameRelayInterconnect(58)	Obsolete use either
aflane8023(59)	ATM Emulated LAN for 802.3
aflane8025(60)	ATM Emulated LAN for 802.5
cctEmul(61)	ATM Emulated circuit
fastEther(62)	Fast Ethernet (100BaseT)
isdn(63)	ISDN and X.25
v11(64)	CCITT V.11/X.21
v36(65)	CCITT V.36
g703at64k(66)	CCITT G703 at 64Kbps
g703at2mb(67)	Obsolete see DS1 MIB
qllc(68)	SNA QLLC
fastEtherFX(69)	Fast Ethernet (100BaseFX)
channel(70)	channel
ieee80211(71)	radio spread spectrum
ibm370parChan(72)	IBM System 360/370 OEMI Channel
escon(73)	IBM Enterprise Systems Connection
dlsW(74)	Data Link Switching
isdns(75)	ISDN S/T interface
isdnu(76)	ISDN U interface
lapd(77)	Link Access Protocol D
ipSwitch(78)	IP Switching Objects
rsrb(79)	Remote Source Route Bridging
atmLogical(80)	ATM Logical Port
ds0(81)	Digital Signal Level 0
ds0Bundle(82)	group of ds0s on the same ds1
bsc(83)	Bisynchronous Protocol
async(84)	Asynchronous Protocol
cnr(85)	Combat Net Radio

iso88025Dtr(86)	ISO 802.5r DTR
eplrs(87)	Ext Pos Loc Report Sys
arap(88)	Appletalk Remote Access Protocol
propCnls(89)	Proprietary Connectionless Protocol
hostPad(90)	CCITT ITU X.29 PAD Protocol
termPad(91)	CCITT ITU X.3 PAD Facility
frameRelayMPI(92)	Multiproto Interconnect over FR
x213(93)	CCITT ITU X213
adsl(94)	Asymmetric Digital Subscriber Loop
radsl(95)	Rate Adapt. Digital Subscriber Loop
sdsl(96)	Symmetric Digital Subscriber Loop
vdsl(97)	Very H Speed Digital Subscrib. Loop
iso88025CRFPInt(98)	ISO 802.5 CRFP
myrinet(99)	Myricom Myrinet
voiceEM(100)	voice receive and transmit
voiceFXO(101)	voice Foreign Exchange Office
voiceFXS(102)	voice Foreign Exchange Station
voiceEncap(103)	voice encapsulation
voiceOverIp(104)	voice over IP encapsulation
atmDxi(105)	ATM DXI
atmFuni(106)	ATM FUNI
atmIma (107)	ATM IMA
pppMultilinkBundle(108)	PPP Multilink Bundle
ipOverCdlc (109)	IBM ipOverCdlc
ipOverClaw (110)	IBM Common Link Access to Workstn
stackToStack (111)	IBM stackToStack
virtualIpAddress (112)	IBM VIPA
mpc (113)	IBM multi protocol channel support
ipOverAtm (114)	IBM ipOverAtm
iso88025Fiber (115)	ISO 802.5j Fiber Token Ring
tdlc (116)	IBM twinaxial data link control
gigabitEthernet (117)	Gigabit Ethernet
hdlc (118)	HDLC
lapf (119)	LAP F
v37 (120)	V.37
x25mlp (121)	Multi Link Protocol
x25huntGroup (122)	X25 Hunt Group
traspHdlc (123)	Transp HDLC
interleave (124)	Interleave channel
fast (125)	Fast channel
ip (126)	IP (for APPN HPR in IP networks)
docsCableMaclayer (127)	CATV Mac Layer
docsCableDownstream (128)	CATV Downstream interface
docsCableUpstream (129)	CATV Upstream interface
a12MppSwitch (130)	Avalon Parallel Processor
tunnel (131)	Encapsulation interface
coffee (132)	coffee pot
ces (133)	Circuit Emulation Service
atmSubInterface (134)	ATM Sub Interface
l2vlan (135)	Layer 2 Virtual LAN using 802.1Q
l3ipvlan (136)	Layer 3 Virtual LAN using IP
l3ipxvlan (137)	Layer 3 Virtual LAN using IPX
digitalPowerline (138)	IP over Power Lines

mediaMailOverIp (139)	Multimedia Mail over IP
dtm (140)	Dynamic synchronous Transfer Mode

## Address Translation Group

Implementation of the Address Translation group is mandatory for all systems. Note however that this group is deprecated by MIB-II. That is, it is being included solely for compatibility with MIB-I nodes, and will most likely be excluded from MIB-III nodes. From MIB-II and onwards, each network protocol group contains its own Address Translation tables.

The Address Translation group contains one table which is the union across all interfaces of the translation tables for converting a Network Address (e.g. an IP address) into a subnetwork-specific address. For lack of a better term, this document refers to such a subnetwork-specific address as a 'physical' address.

Examples of such translation tables are: for broadcast media where ARP is in use, the translation table is equivalent to the ARP cache; or, on an X.25 network where non-algorithmic translation to X.121 addresses is required, the translation table contains the NetworkAddress to X.121 address equivalences.

Its OID is *mib.3*.

Object Name	SubID	Access	Description
atTable	3	not-accessible	The Address Translation tables contain the Network Address to 'physical' address equivalences. Some interfaces do not use translation tables for determining address equivalences (e.g. DDN-X.25 has an algorithmic method); if all interfaces are of this type, then the Address Translation table is empty, i.e. has zero entries.

atEntry	3	not-accessible	Each entry contains one NetworkAddress to 'physical' address equivalence.
---------	---	----------------	---

atIfIndex	3.1.1.1	read-write	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.
atPhysAddress	3.1.1.2	read-write	The media-dependent 'physical' address. Setting this object to a null string (one of zero length) has the effect of invalidating the corresponding entry in the atTable object. That is, it effectively disassociates the interface identified with said entry from the

			mapping identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant <code>atPhysAddress</code> object.
<code>atNetAddress</code>	3.1.1.3	read-write	The Network Address (e.g. the IP address) corresponding to the media-dependent 'physical' address.

## ***IP Group***

This is a collection of IP information for the device.

Implementation of the IP group is mandatory for all systems.

Its OID is *mib.4*.

<b>Object Name</b>	<b>SubID</b>	<b>Access</b>	<b>Description</b>
<code>ipForwarding</code>	1	read-write	The indication of whether this entity is acting as an IP gateway in respect to the forwarding of datagrams received by, but not addressed to, this entity. IP gateways forward datagrams. IP hosts do not (except those source-routed via the host). Note that for some managed nodes, this object may take on only a subset of the values possible. Accordingly, it is appropriate for an agent to return a 'badValue' response if a management station attempts to change this object to an inappropriate value.
<code>ipDefaultTTL</code>	2	read-write	The default value inserted into the Time-To-Live field of the IP header of datagrams originated at this entity, whenever a TTL value is not supplied by the transport layer protocol.
<code>ipInReceives</code>	3	read-only	The total number of input datagrams received from interfaces, including those received in error.
<code>ipInHdrErrors</code>	4	read-only	The number of input datagrams discarded due to errors in their IP headers, including bad checksums, version number mismatch, other format errors,

			time-to-live exceeded, errors discovered in processing their IP options, etc.
ipInAddrErrors	5	read-only	The number of input datagrams discarded because the IP address in their IP header's destination field was not a valid address to be received at this entity. This count includes invalid addresses (e.g. 0.0.0.0) and addresses of unsupported Classes (e.g. Class E). For entities which are not IP Gateways and therefore do not forward datagrams, this counter includes datagrams discarded because the destination address was not a local address.
ipForwDatagrams	6	read-only	The number of input datagrams for which this entity was not their final IP destination, as a result of which an attempt was made to find a route to forward them to that final destination. In entities which do not act as IP Gateways, this counter will include only those packets which were Source-Routed via this entity, and the Source-Route option processing was successful.
ipInUnknownProtos	7	read-only	The number of locally-addressed datagrams received successfully but discarded because of an unknown or unsupported protocol.
ipInDiscards	8	read-only	The number of input IP datagrams for which no problems were encountered to prevent their continued processing, but which were discarded (e.g. for lack of buffer space). Note that this counter does not include any datagrams discarded while awaiting re-assembly.
ipInDelivers	9	read-only	The total number of input datagrams successfully delivered to IP user-protocols (including ICMP).
ipOutRequests	10	read-only	The total number of IP datagrams which local IP user-protocols (including ICMP) supplied to IP in requests for transmission. Note that this counter does not include any datagrams counted in ipForwDatagrams.
ipOutDiscards	11	read-only	The number of output IP datagrams for which no problem was encountered to prevent their transmission to their destination, but which were discarded (e.g. for lack of buffer space). Note that this counter would include datagrams counted in ipForwDatagrams if any such packets met this (discretionary) discard criterion.
ipOutNoRoutes	12	read-only	The number of IP datagrams discarded

			because no route could be found to transmit them to their destination. Note that this counter includes any packets counted in ipForwDatagrams which meet this 'no-route' criterion. Note that this includes any datagrams which a host cannot route because all of its default gateways are down.
ipReasmTimeout	13	read-only	The maximum number of seconds which received fragments are held while they are awaiting reassembly at this entity.
ipReasmReqds	14	read-only	The number of IP fragments received which needed to be reassembled at this entity.
ipReasmOKs	15	read-only	The number of IP datagrams successfully re-assembled.
ipReasmFails	16	read-only	The number of failures detected by the IP re-assembly algorithm (for whatever reason: timed out, errors, etc). Note that this is not necessarily a count of discarded IP fragments since some algorithms (notably the algorithm in RFC 815) can lose track of the number of fragments by combining them as they are received.
ipFragOKs	17	read-only	The number of IP datagrams that have been successfully fragmented at this entity.
ipFragFails	18	read-only	The number of IP datagrams that have been discarded because they needed to be fragmented at this entity but could not be, e.g. because their Don't Fragment flag was set.
ipFragCreates	19	read-only	The number of IP datagram fragments that have been generated as a result of fragmentation at this entity.
ipAddrTable	20	not-accessible	The table of addressing information relevant to this entity's IP addresses.  This table is expanded below.
ipRouteTable	21	not-accessible	This entity's IP Routing table.  This table is expanded below.
ipNetToMediaTable	22	not-accessible	The IP Address Translation table used for mapping from IP addresses to physical addresses.  This table is expanded below.
ipRoutingDiscards	23	read-only	The number of routing entries which were chosen to be discarded even though they are valid. One possible reason for discarding such an entry could be to free-up buffer space for other routing entries.

The *ipAddrTable* contains this entity's IP addressing information.

Object Name	SubID	Access	Description
ipAddrEntry	20	not-accessible	The addressing information for one of this entity's IP addresses.

ipAdEntAddr	20.1.1	read-only	The IP address to which this entry's addressing information pertains.
ipAdEntIfIndex	20.1.2	read-only	The index value which uniquely identifies the interface to which this entry is applicable. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex.
ipAdEntNetMask	20.1.3	read-only	The subnet mask associated with the IP address of this entry. The value of the mask is an IP address with all the network bits set to 1 and all the hosts bits set to 0.
ipAdEntBcastAddr	20.1.4	read-only	The value of the least-significant bit in the IP broadcast address used for sending datagrams on the (logical) interface associated with the IP address of this entry. For example, when the Internet standard all-ones broadcast address is used, the value will be 1. This value applies to both the subnet and network broadcasts addresses used by the entity on this (logical) interface.
ipAdEntReasmMaxSize	20.1.5	read-only	The size of the largest IP datagram which this entity can re-assemble from incoming IP fragmented datagrams received on this interface.

The *ipRouteTable* contains an entry for each route presently known to this entity.

Object Name	SubID	Access	Description
ipRouteEntry	21	not-accessible	A route to a particular destination.

ipRouteDest	21.1.1	read-write	The destination IP address of this route. An entry with a value of 0.0.0.0 is considered a default route. Multiple routes to a single destination can appear in the table, but access to such multiple entries is dependent on the table- access mechanisms defined by the network
-------------	--------	------------	--

			management protocol in use.
ipRouteIfIndex	21.1.2	read-write	The index value which uniquely identifies the local interface through which the next hop of this route should be reached. The interface identified by a particular value of this index is the same interface as identified by the same value of ifIndex.
ipRouteMetric1	21.1.3	read-write	The primary routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.
ipRouteMetric2	21.1.4	read-write	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.
ipRouteMetric3	21.1.5	read-write	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.
ipRouteMetric4	21.1.6	read-write	An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.
ipRouteNextHop	21.1.7	read-write	The IP address of the next hop of this route. (In the case of a route bound to an interface which is realised via a broadcast media, the value of this field is the agent's IP address on that interface.)
ipRouteType	21.1.8	read-write	The type of route. Note that the values direct(3) and indirect(4) refer to the notion of direct and indirect routing in the IP architecture. Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the ipRouteTable object. That is, it effectively disassociates the destination identified with said entry from the route identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such

			<p>entries requires examination of the relevant ipRouteType object.</p> <p>Values are:</p> <p>other(1), none of the following  invalid(2), an invalidated route  direct(3), route to directly connected (sub-)network  indirect(4) route to a non-local host/network/sub-network</p>
ipRouteProto	21.1.9	read-only	<p>The routing mechanism via which this route was learned. Inclusion of values for gateway routing protocols is not intended to imply that hosts should support those protocols</p> <p>Values are:</p> <p>other(1) none of the following  local(2) non-protocol information, e.g. manually configured entries  netmgmt(3) set via a network management protocol  icmp(4) obtained via ICMP, e.g. Redirect  egp(5)  ggp(6)  hello(7)  rip(8)  is-is(9)  es-is(10)  ciscoIgrp(11)  bbnSpfIgp(12)  ospf(13)  bgp(14)</p>
ipRouteAge	21.1.10	read-write	<p>The number of seconds since this route was last updated or otherwise determined to be correct. Note that no semantics of 'too old' can be implied except through knowledge of the routing protocol by which the route was learned.</p>
ipRouteMask	21.1.11	read-write	<p>Indicate the mask to be logical-AND'd with the destination address before being compared to the value in the ipRouteDest field. For those systems that do not support arbitrary subnet masks, an agent constructs the value of the ipRouteMask by determining whether the value of the correspondent ipRouteDest field belong to a class-A, B, or C network, and then using one of:</p> <p><u>mask</u>                      <u>network</u>  255.0.0.0                      class-A</p>

			<p>255.255.0.0 class-B 255.255.255.0 class-C</p> <p>If the value of the ipRouteDest is 0.0.0.0 (a default route), then the mask value is also 0.0.0.0. It should be noted that all IP routing subsystems implicitly use this mechanism.</p>
ipRouteMetric5	21.1.12	read-write	<p>An alternate routing metric for this route. The semantics of this metric are determined by the routing-protocol specified in the route's ipRouteProto value. If this metric is not used, its value should be set to -1.</p>
ipRouteInfo	21.1.13	read-only	<p>A reference to MIB definitions specific to the particular routing protocol which is responsible for this route, as determined by the value specified in the route's ipRouteProto value. If this information is not present, its value should be set to the OBJECT IDENTIFIER { 0 0 }, which is a syntatically valid object identifier, and any conformant implementation of ASN.1 and BER must be able to generate and recognise this value.</p>

The *ipNetToMediaTable* contains the IpAddress to 'physical' address equivalences. Some interfaces do not use translation tables for determining address equivalences (e.g. DDN-X.25 has an algorithmic method). If all interfaces are of this type, then the AddressTranslation table is empty, i.e. has zero entries.

Object Name	SubID	Access	Description
ipNetToMediaEntry	22	not-accessible	Each entry contains one IpAddress to 'physical' address equivalence.

ipNetToMediaIfIndex	22.1.1	read-write	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.
ipNetToMediaPhysAddress	22.1.2	read-write	The media-dependent 'physical' address.
ipNetToMediaNetAddress	22.1.3	read-write	The IpAddress corresponding to the media-dependent 'physical' address.
ipNetToMediaType	22.1.4	read-write	The type of mapping. Setting this object to the value invalid(2) has the effect of invalidating the corresponding entry in the ipNetToMediaTable. That is, it effectively disassociates the interface

			identified with said entry from the mapping identified with said entry. It is an implementation-specific matter as to whether the agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the relevant ipNetToMediaType object.
--	--	--	---

## ICMP Group

This is a collection of ICMP information for the device.

Implementation of the ICMP group is mandatory for all systems.

Its OID is *mib.5*.

Object Name	SubID	Access	Description
icmpInMsgs	1	read-only	The total number of ICMP messages which the entity received. Note that this counter includes all those counted by icmpInErrors.
icmpInErrors	2	read-only	The number of ICMP messages which the entity received but determined as having ICMP-specific errors (bad ICMP checksums, bad length, etc.).
icmpInDestUnreachs	3	read-only	The number of ICMP Destination Unreachable messages received.
icmpInTimeExcds	4	read-only	The number of ICMP Time Exceeded messages received.
icmpInParmProbs	5	read-only	The number of ICMP Parameter Problem messages received.
icmpInSrcQuenchs	6	read-only	The number of ICMP Source Quench messages received.
icmpInRedirects	7	read-only	The number of ICMP Redirect messages received.
icmpInEchos	8	read-only	The number of ICMP Echo (request) messages received.
icmpInEchoReps	9	read-only	The number of ICMP Echo Reply messages received.
icmpInTimestamps	10	read-only	The number of ICMP Timestamp (request) messages received.
icmpInTimestampReps	11	read-only	The number of ICMP Timestamp Reply messages received.
icmpInAddrMasks	12	read-only	The number of ICMP Address Mask Request messages received.
icmpInAddrMaskReps	13	read-only	The number of ICMP Address Mask Reply

			messages received.
icmpOutMsgs	14	read-only	The total number of ICMP messages which this entity attempted to send. Note that this counter includes all those counted by icmpOutErrors.
icmpOutErrors	15	read-only	The number of ICMP messages which this entity did not send due to problems discovered within ICMP such as a lack of buffers. This value should not include errors discovered outside the ICMP layer such as the inability of IP to route the resultant datagram. In some implementations there may be no types of error which contribute to this counter's value.
icmpOutDestUnreachs	16	read-only	The number of ICMP Destination Unreachable messages sent.
icmpOutTimeExcds	17	read-only	The number of ICMP Time Exceeded messages sent.
icmpOutParmProbs	18	read-only	The number of ICMP Parameter Problem messages sent.
icmpOutSrcQuenchs	19	read-only	The number of ICMP Source Quench messages sent.
icmpOutRedirects	20	read-only	The number of ICMP Redirect messages sent. For a host, this object will always be zero, since hosts do not send redirects.
icmpOutEchos	21	read-only	The number of ICMP Echo (request) messages sent.
icmpOutEchoReps	22	read-only	The number of ICMP Echo Reply messages sent.
icmpOutTimestamps	23	read-only	The number of ICMP Timestamp (request) messages sent.
icmpOutTimestampReps	24	read-only	The number of ICMP Timestamp Reply messages sent.
icmpOutAddrMasks	25	read-only	The number of ICMP Address Mask Request messages sent.
icmpOutAddrMaskReps	26	read-only	The number of ICMP Address Mask Reply messages sent.

## **TCP Group**

This is a collection of information about transient TCP connections

Implementation of the TCP group is mandatory for all systems.

Its OID is *mib.6*.

Object Name	SubID	Access	Description
tcpRtoAlgorithm	1	read-only	The algorithm used to determine the timeout value used for retransmitting unacknowledged octets.
tcpRtoMin	2	read-only	The minimum value permitted by a TCP implementation for the retransmission timeout, measured in milliseconds. More refined semantics for objects of this type depend upon the algorithm used to determine the retransmission timeout. In particular, when the timeout algorithm is rsre(3), an object of this type has the semantics of the LBOUND quantity described in RFC 793.
tcpRtoMax	3	read-only	The maximum value permitted by a TCP implementation for the retransmission timeout, measured in milliseconds. More refined semantics for objects of this type depend upon the algorithm used to determine the retransmission timeout. In particular, when the timeout algorithm is rsre(3), an object of this type has the semantics of the UBOUND quantity described in RFC 793.
tcpMaxConn	4	read-only	The limit on the total number of TCP connections the entity can support. In entities where the maximum number of connections is dynamic, this object should contain the value -1.
tcpActiveOpens	5	read-only	The number of times TCP connections have made a direct transition to the SYN-SENT state from the CLOSED state.
tcpPassiveOpens	6	read-only	The number of times TCP connections have made a direct transition to the SYN-RCVD state from the LISTEN state.
tcpAttemptFails	7	read-only	The number of times TCP connections have made a direct transition to the CLOSED state from either the SYN-SENT state or the SYN-RCVD state, plus the number of times TCP connections have made a direct transition to the LISTEN state from the SYN-RCVD state.
tcpEstabResets	8	read-only	The number of times TCP connections have made a direct transition to the CLOSED state from either the ESTABLISHED state or the CLOSE-WAIT state.
tcpCurrEstab	9	read-only	The number of TCP connections for which the current state is either

			ESTABLISHED or CLOSE-WAIT.
tcpInSegs	10	read-only	The total number of segments received, including those received in error. This count includes segments received on currently established connections.
tcpOutSegs	11	read-only	The total number of segments sent, including those on current connections but excluding those containing only retransmitted octets.
tcpRetransSegs	12	read-only	The total number of segments retransmitted - that is, the number of TCP segments transmitted containing one or more previously transmitted octets.
tcpConnTable	13	not-accessible	A table containing TCP connection-specific information.  This table is expanded below.
tcpInErrs	14	read-only	The total number of segments received in error (e.g. bad TCP checksums).
tcpOutRsts	15	read-only	The number of TCP segments sent containing the RST flag.

The *tcpConnTable* contains information about this entity's existing TCP connections.

Object Name	SubID	Access	Description
tcpConnEntry	13	not-accessible	Information about a particular current TCP connection. An object of this type is transient, in that it ceases to exist when (or soon after) the connection makes the transition to the CLOSED state.

tcpConnState	13.1.1	read-write	The state of this TCP connection. The only value which may be set by a management station is deleteTCB(12). Accordingly, it is appropriate for an agent to return a 'badValue' response if a management station attempts to set this object to any other value. If a management station sets this object to the value deleteTCB(12), then this has the effect of deleting the TCB (as defined in RFC 793) of the corresponding connection on the managed node, resulting in immediate termination of the connection. As an implementation-specific option, a RST segment may be sent from the managed node to the other TCP endpoint (note however that RST segments are not
--------------	--------	------------	--

			sent reliably).
tcpConnLocalAddress	13.1.2	read-only	The local IP address for this TCP connection. In the case of a connection in the listen state which is willing to accept connections for any IP interface associated with the node, the value 0.0.0.0 is used.
tcpConnLocalPort	13.1.3	read-only	The local port number for this TCP connection.
tcpConnRemAddress	13.1.4	read-only	The remote IP address for this TCP connection.
tcpConnRemPort	13.1.5	read-only	The remote port number for this TCP connection.

## UDP Group

This is a collection of UPD information for the device.

Implementation of the UDP group is mandatory for all systems which implement the UDP.

Its OID is *mib.7*.

Object Name	SubID	Access	Description
udpInDatagrams	1	read-only	The total number of UDP datagrams delivered to UDP users.
udpNoPorts	2	read-only	The total number of received UDP datagrams for which there was no application at the destination port.
udpInErrors	3	read-only	The number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port.
udpOutDatagrams	4	read-only	The total number of UDP datagrams sent from this entity.
udpTable	5	not-accessible	A table containing UDP listener information.  This table is expanded below.

The *udpTable* contains information about this entity's UDP end-points on which a local application is currently accepting datagrams.

Object Name	SubID	Access	Description
udpEntry	5	not-accessible	Information about a particular current UDP listener.

udpLocalAddress	5.1.1	read-only	The local IP address for this UDP listener. In the case of a UDP listener which is willing to accept datagrams for any IP interface associated with the node, the value 0.0.0.0 is used.
udpLocalPort	5.1.2	read-only	The local port number for this UDP listener.

## **EGP Group**

This is a collection of EGP information for the device.

Implementation of the EGP group is mandatory for all systems which implement the EGP.

Its OID is *mib.8*.

<b>Object Name</b>	<b>SubID</b>	<b>Access</b>	<b>Description</b>
egpInMsgs	1	read-only	The number of EGP messages received without error.
egpInErrors	2	read-only	The number of EGP messages received that proved to be in error.
egpOutMsgs	3	read-only	The total number of locally generated EGP messages.
egpOutErrors	4	read-only	The number of locally generated EGP messages not sent due to resource limitations within an EGP entity.
egpNeighTable	5	not-accessible	The EGP neighbour table.  This table is expanded below.
egpAs	6	read-only	The autonomous system number of this EGP entity.

The *egpNeighTable* contains information about this entity's EGP neighbours.

<b>Object Name</b>	<b>SubID</b>	<b>Access</b>	<b>Description</b>
egpNeighEntry	5	not-accessible	Information about this entity's relationship with a particular EGP neighbour.

egpNeighState	5.1.1	read-only	The EGP state of the local system with respect to this entry's EGP neighbour. Each EGP state is represented by a value that is one greater than the
---------------	-------	-----------	---

			numerical value associated with said state in RFC 904.
egpNeighAddr	5.1.2	read-only	The IP address of this entry's EGP neighbour.
egpNeighAs	5.1.3	read-only	The autonomous system of this EGP peer. Zero should be specified if the autonomous system number of the neighbour is not yet known.
egpNeighInMsgs	5.1.4	read-only	The number of EGP messages received without error from this EGP peer.
egpNeighInErrs	5.1.5	read-only	The number of EGP messages received from this EGP peer that proved to be in error (e.g. bad EGP checksum).
egpNeighOutMsgs	5.1.6	read-only	The number of locally generated EGP messages to this EGP peer.
egpNeighOutErrs	5.1.7	read-only	The number of locally generated EGP messages not sent to this EGP peer due to resource limitations within an EGP entity.
egpNeighInErrMsgs	5.1.8	read-only	The number of EGP-defined error messages received from this EGP peer.
egpNeighOutErrMsgs	5.1.9	read-only	The number of EGP-defined error messages sent to this EGP peer.
egpNeighStateUps	5.1.10	read-only	The number of EGP state transitions to the UP state with this EGP peer.
egpNeighStateDowns	5.1.11	read-only	The number of EGP state transitions from the UP state to any other state with this EGP peer.
egpNeighIntervalHello	5.1.12	read-only	The interval between EGP Hello command retransmissions (in hundredths of a second). This represents the t1 timer as defined in RFC 904.
egpNeighIntervalPoll	5.1.13	read-only	The interval between EGP poll command retransmissions (in hundredths of a second). This represents the t3 timer as defined in RFC 904.
egpNeighMode	5.1.14	read-only	The polling mode of this EGP entity, either passive or active.
egpNeighEventTrigger	5.1.15	read-write	A control variable used to trigger operator- initiated Start and Stop events. When read, this variable always returns the most recent value that egpNeighEventTrigger was set to. If it has not been set since the last initialisation of the network management subsystem on the node, it returns a value of `stop'. When set, this variable causes a Start or Stop event on the specified neighbour, as specified on pages 8-10 of RFC 904. Briefly, a Start event causes an Idle peer to begin neighbour acquisition and a non-Idle

			peer to reinitiate neighbour acquisition. A stop event causes a non-Idle peer to return to the Idle state until a Start event occurs, either via egpNeighEventTrigger or otherwise.
--	--	--	---

## SNMP Group

This is a collection of SNMP information for the device.

Implementation of the SNMP group is mandatory for all systems which support an SNMP protocol entity. Some of the objects defined below will be zero-valued in those SNMP implementations that are optimised to support only those functions specific to either a management agent or a management station. In particular, it should be observed that the objects below refer to an SNMP entity, and there may be several SNMP entities residing on a managed node (e.g. if the node is hosting acting as a management station).

Its OID is *mib.11*.

Object Name	SubID	Access	Description
snmpInPkts	1	read-only	The total number of Messages delivered to the SNMP entity from the transport service.
snmpOutPkts	2	read-only	The total number of SNMP Messages which were passed from the SNMP protocol entity to the transport service.
snmpInBadVersions	3	read-only	The total number of SNMP Messages which were delivered to the SNMP protocol entity and were for an unsupported SNMP version.
snmpInBadCommunityNames	4	read-only	The total number of SNMP Messages delivered to the SNMP protocol entity which used a SNMP community name not known to said entity.
snmpInBadCommunityUses	5	read-only	The total number of SNMP Messages delivered to the SNMP protocol entity which represented an SNMP operation which was not allowed by the SNMP community named in the Message.
snmpInASNParseErrs	6	read-only	The total number of ASN.1 or BER errors encountered by the SNMP protocol entity when decoding received SNMP Messages.
snmpInTooBig	8	read-only	The total number of SNMP PDUs which were delivered to the SNMP

			protocol entity and for which the value of the error-status field is 'tooBig'.
snmpInNoSuchNames	9	read-only	The total number of SNMP PDUs which were delivered to the SNMP protocol entity and for which the value of the error-status field is 'noSuchName'.
snmpInBadValues	10	read-only	The total number of SNMP PDUs which were delivered to the SNMP protocol entity and for which the value of the error-status field is 'badValue'.
snmpInReadOnlys	11	read-only	The total number valid SNMP PDUs which were delivered to the SNMP protocol entity and for which the value of the error-status field is 'readOnly'. It should be noted that it is a protocol error to generate an SNMP PDU which contains the value 'readOnly' in the error-status field, as such this object is provided as a means of detecting incorrect implementations of the SNMP.
snmpInGenErrs	12	read-only	The total number of SNMP PDUs which were delivered to the SNMP protocol entity and for which the value of the error-status field is 'genErr'.
snmpInTotalReqVars	13	read-only	The total number of MIB objects which have been retrieved successfully by the SNMP protocol entity as the result of receiving valid SNMP Get-Request and Get-Next PDUs.
snmpInTotalSetVars	14	read-only	The total number of MIB objects which have been altered successfully by the SNMP protocol entity as the result of receiving valid SNMP Set-Request PDUs.
snmpInGetRequests	15	read-only	The total number of SNMP Get-Request PDUs which have been accepted and processed by the SNMP protocol entity.
snmpInGetNexts	16	read-only	The total number of SNMP Get-Next PDUs which have been accepted and processed by the SNMP protocol entity.
snmpInSetRequests	17	read-only	The total number of SNMP Set-Request PDUs which have been accepted and processed by the SNMP protocol entity.
snmpInGetResponses	18	read-only	The total number of SNMP Get-Response PDUs which have been

			accepted and processed by the SNMP protocol entity.
snmpInTraps	19	read-only	The total number of SNMP Trap PDUs which have been accepted and processed by the SNMP protocol entity.
snmpOutTooBig	20	read-only	The total number of SNMP PDUs which were generated by the SNMP protocol entity and for which the value of the error-status field is 'tooBig.'
snmpOutNoSuchNames	21	read-only	The total number of SNMP PDUs which were generated by the SNMP protocol entity and for which the value of the error-status is 'noSuchName'.
snmpOutBadValues	22	read-only	The total number of SNMP PDUs which were generated by the SNMP protocol entity and for which the value of the error-status field is 'badValue'.
snmpOutGenErrs	24	read-only	The total number of SNMP PDUs which were generated by the SNMP protocol entity and for which the value of the error-status field is 'genErr'.
snmpOutGetRequests	25	read-only	The total number of SNMP Get-Request PDUs which have been generated by the SNMP protocol entity.
snmpOutGetNexts	26	read-only	The total number of SNMP Get-Next PDUs which have been generated by the SNMP protocol entity.
snmpOutSetRequests	27	read-only	The total number of SNMP Set-Request PDUs which have been generated by the SNMP protocol entity.
snmpOutGetResponses	28	read-only	The total number of SNMP Get-Response PDUs which have been generated by the SNMP protocol entity.
snmpOutTraps	28	read-only	The total number of SNMP Trap PDUs which have been generated by the SNMP protocol entity.
snmpEnableAuthenTraps	30	read-write	Indicates whether the SNMP agent process is permitted to generate authentication-failure traps. The value of this object overrides any configuration information; as such, it provides a means whereby all authentication-failure traps may be disabled. Note that it is strongly recommended that this object be stored in non-volatile memory so that

			it remains constant between re-initialisations of the network management system.
snmpSilentDrops	31	read-only	The total number of GetRequest-PDUs, GetNextRequest-PDUs, GetBulkRequest-PDUs, SetRequest-PDUs, and InformRequest-PDUs delivered to the SNMP entity which were silently dropped because the size of a reply containing an alternate Response-PDU with an empty variable-bindings field was greater than either a local constraint or the maximum message size associated with the originator of the request.
snmpProxyDrops	32	read-only	The total number of GetRequest-PDUs, GetNextRequest-PDUs, GetBulkRequest-PDUs, SetRequest-PDUs, and InformRequest-PDUs delivered to the SNMP entity which were silently dropped because the transmission of the (possibly translated) message to a proxy target failed in a manner (other than a time-out) such that no Response-PDU could be returned.