Chapter 4

SNMPv1 Network Management:

Organization and Information Models

Assigned Readings:

- Chapter 4 in Subramanian, Gonsalves & Rani (2010) – All sections

Objectives

- IETF SNMP standard
  - **History**
  - **RFC, STD, and FYI**
- **Organization** Model
  - 2- and 3-tier models
  - Manager and agent
- Management messages
- Structure of management information, **SMI**
- Object type and instance
- Scalar and aggregate managed objects
- Management information base, **MIB**
- NMS physical and virtual databases
- IETF MIB-2 standard

Case Histories

- AT&T Network Management Centers
  - Network Control Centers
  - Network Operations Center
- CNN World Headquarters
- Centralized troubleshooting of NIC
- Performance degradation due to NMS
- Bell Operating company procedure

Notes

Several visits show how big corporations and institutions manage their big networks

- **Automated network monitoring with network and status shown on large screen**
- **Automated Recovery from failure for a wide range of failures**
- **alarms for unrecoverable failures**
- **remote tests for network parts**
- **………**

Managed LAN: example

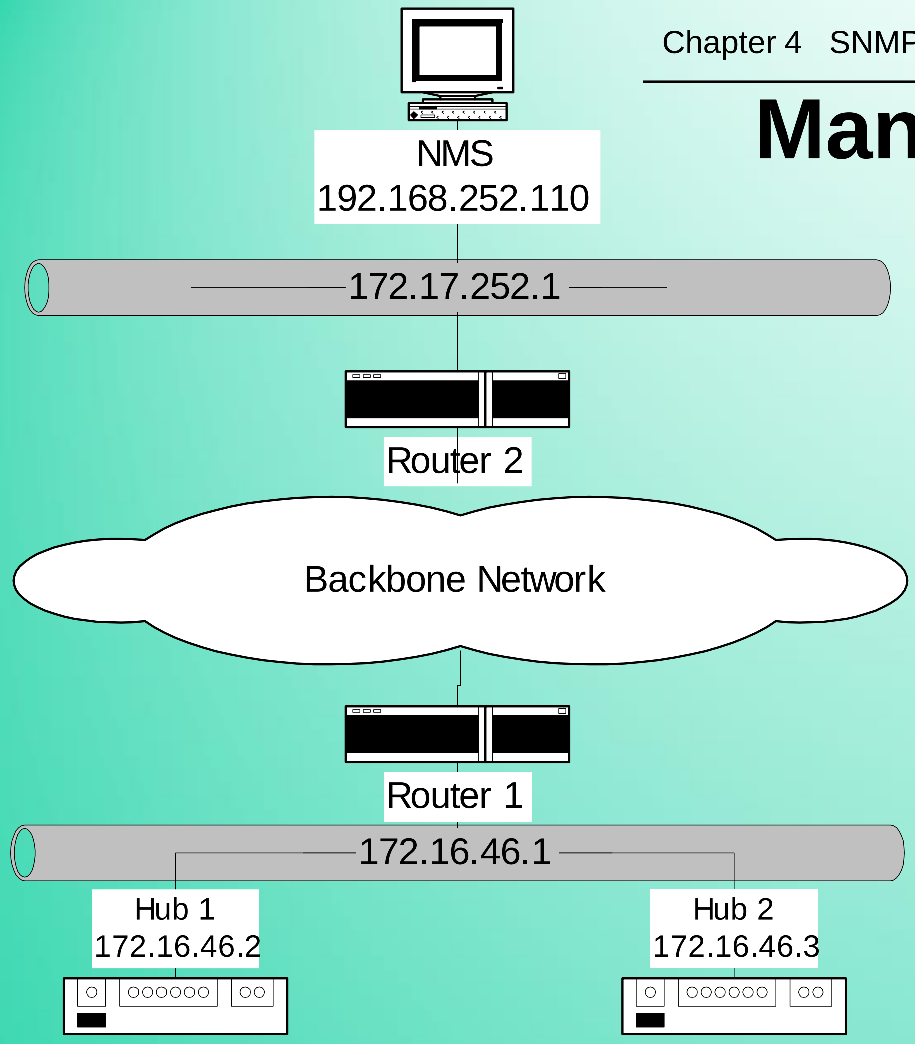


Figure 4.1 Managed LAN Network

A backbone is a larger transmission line that carries data gathered from smaller lines that interconnect with it.

- NMS on subnet 192.168.252.1 manages the router and the hubs on subnet 172.16.46.1 across the backbone network

The NMS, whose IP address is 192.168.252.110, is physically and logically located remotely from the 172.16.46.1 LAN. It is configured on the LAN 192.168.252.1 and is connected to the backbone network.

Managed Hub (Hub 1): System Information

Title: System Information: 172.16.46.2
Name or IP Address: 172.16.46.2

System Name:
System Description: 3Com LinkBuilder FMS, SW version:3.02
System Contact:
System Location:
System Object ID: iso.org.dod.internet.private.enterprises.43.1.8.5
System Up Time: (2475380437) 286 days, 12:03:24.37

Figure 4.2(a) System Information on 172.16.46.2 Hub

Notes

- Information obtained by querying the hub
- Data truly reflects what is stored in the hub

Managed Router (router 2): System Information

Title: System Information: router1.gatech.edu
Name or IP Address: 172.16.252.1

System Name	:	router1.gatech.edu
System Description	:	Cisco Internetwork Operating System Software
	:	IOS (tm) 7000 Software (C7000-JS-M), Version
	:	11.2(6),RELEASE SOFTWARE (ge1)
	:	Copyright (c) 1986-1997 by Cisco Systems, Inc.
	:	Compiled Tue 06-May-97 19:11 by kuong
System Contact		
System Location	:	
System Object ID	:	iso.org.dod.internet.private.enterprises.cisco.ciscoProducts. cisco 7000
System Up Time	:	(315131795) 36 days, 11:21:57.95

Figure 4.2(c) System Information on Router

Notes

Managed Hub: Port Addresses

Index	Interface	IP address	Network Mask	Network Address	Link Address
1	3Com	172.16.46.2	255.255.255.0	172.16 46.0	0x08004E07C25C
2	3Com	192.168.101.1	255.255.255.0	192.168.101.0	<none>

Notes

- Information acquired by the NMS on hub interfaces
- Index refers to the interface on the hub
- Link address is the MAC address
- The second row data is a serial link (serial port hub)

Managed Router: Port Addresses

Index	Interface	IP address	Network Mask	Network Address	Link Address
23	LEC.1.0	192.168.3.1	255.255.255.0	192.168.3.0	0x00000C3920B4
25	LEC.3.9	192.168.252.15	255.255.255.0	192.168.252.0	0x00000C3920B4
13	Ethernet2/0	172.16..46.1	255.255.255.0	172.16..46.0	0x00000C3920AC
16	Ethernet2/3	172.16.49.1	255.255.255.0	172.16.49.0	0x00000C3920AF
17	Ethernet2/4	172.16.52.1	255.255.255.0	172.16.52.0	0x00000C3920B0
9	Ethernet1/2	172.16.55.1	255.255.255.0	172.16.55.0	0x00000C3920A6
2	Ethernet 0/1	172.16.56.1	255.255.255.0	172.16.56.0	0x00000C39209D
15	Ethernet2/2	172.16.57.1	255.255.255.0	172.16.57.0	0x00000C3920AE
8	Ethernet1/1	172.16.58.1	255.255.255.0	172.16.58.0	0x00000C3920A5
14	Ethernet2/1	172.16.60.1	255.255.255.0	172.16.60.0	0x00000C3920AD

Notes

- Information acquired by NMS on the router interfaces
- Index refers to the interface on the router
- LEC is the LAN emulation card (for interface with ATM networks)
- Ethernet 2/0 interface refers to the interface card 2 and port 0 in that card

History of Internet SNMP Management

- 1970s Advanced Research Project Agency Network (ARPANET) Internet Control Message Protocol (ICMP)

- Internet Engineering Task Force (IETF)
  - 1990 SNMPv1
  - 1995 SNMPv2
  - 1998 SNMPv3
- Internet documents:
  - Request for Comments (**RFC**)
    - IETF **STD** Internet Standard: (standards)
    - **FYI** For Your Information: documents overviews and introductory topics

- Source for RFCs
  - ftp://nic.mil/rfc
  - ftp://ftp.internic.net/rfc
  - http://nic/internet.net/

Notes

SNMPv1 & SNMPv2 Documents

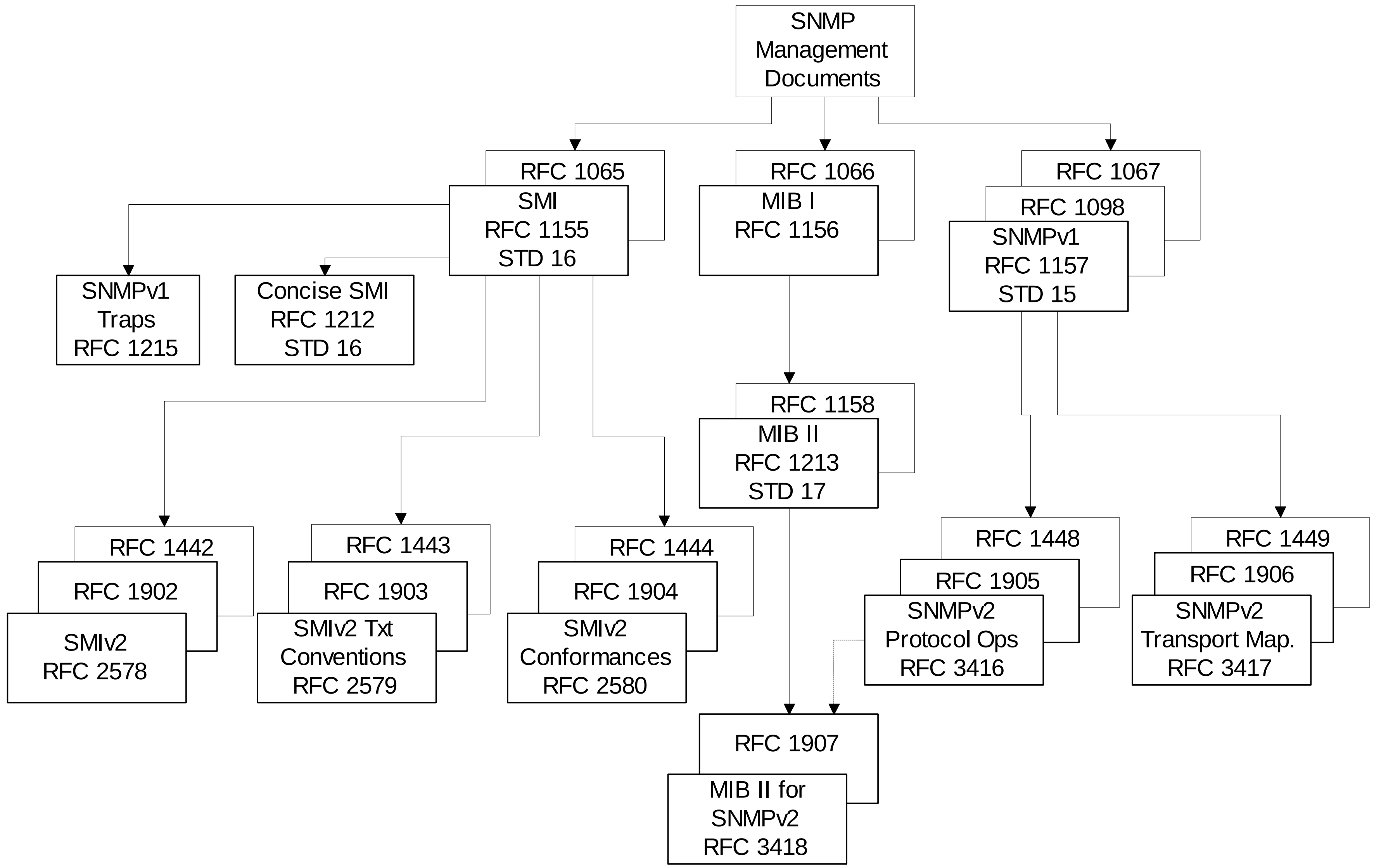


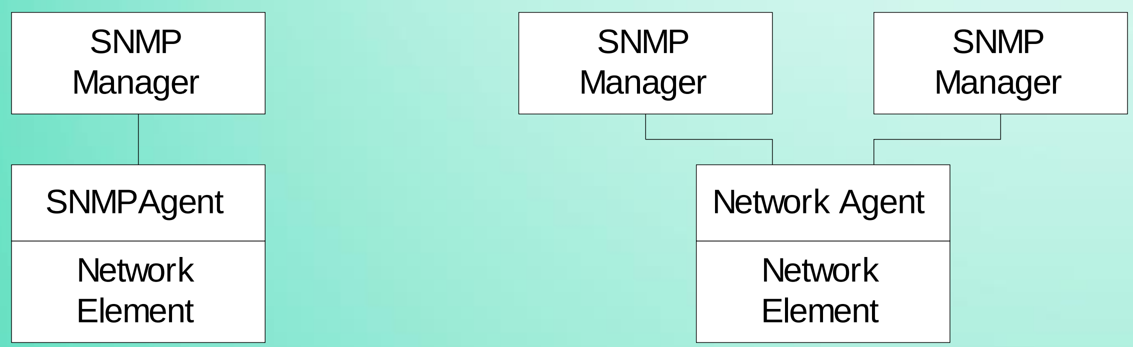
Figure 4.4 SNMP Document Evolution

Notes

SNMP Model

- Organization Model
  - Relationship between network element, agent, and manager
  - Hierarchical architecture
- Information Model
  - Uses ASN.1 syntax
  - SMI (Structure of Management Information)
  - MIB (Management Information Base)
- Communication Model
  - Transfer syntax
  - SNMP over TCP/IP
  - Communication services addressed by messages
  - Security framework community-based model
- Functional model
  - Fault management
  - Configuration management
  - Account management
  - Performance management
  - Security management

Two-Tier Organization Model



(a) One Manager-One Agent Model **(b) Multiple Managers-One Agent Model**

Figure 4.5 Two-Tier Organization Model

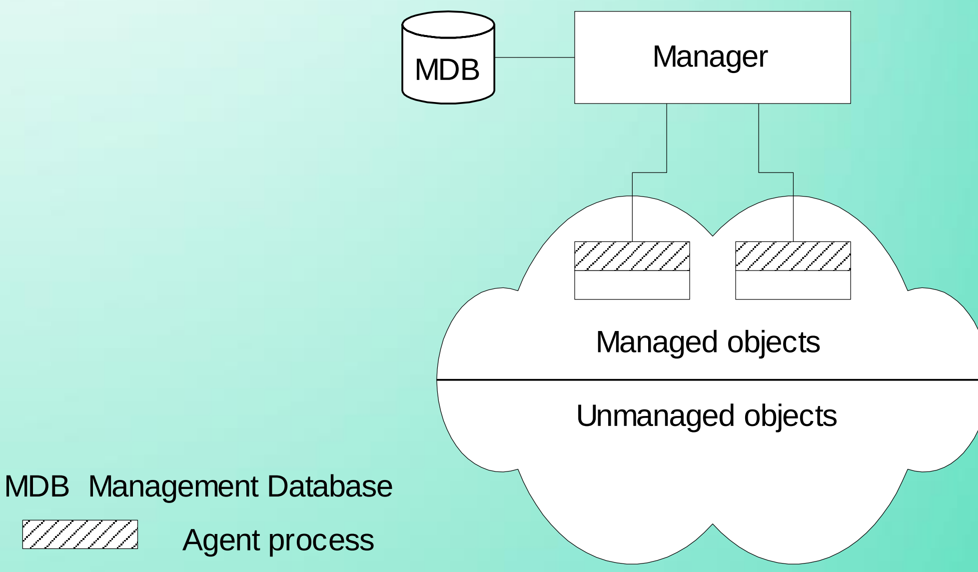


Figure 3.2 Two-Tier Network Management Organization Model

Notes

- Any host that could query an agent is a manager.

The initial organization model of SNMP management is a simple two-tier model. It consists of a network agent process, which resides in the managed object, and a network manager process, which resides in the NMS and manages the managed object. This is shown in Figure 4.5(a). Both the manager and the agent are software modules. The agent responds to any management system that communicates with it using SNMP. Thus, multiple managers can interact with one agent as shown in Figure 4.5(b)

Three-Tier Organization Model: RMON

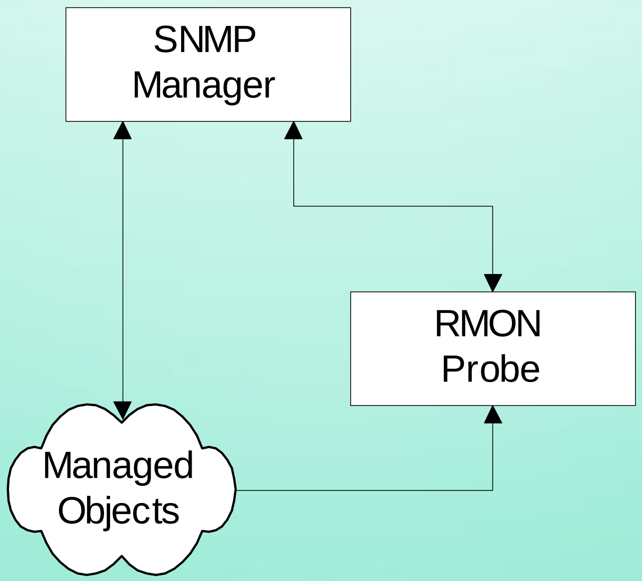


Figure 4.6 Three-Tier Organization Model

Notes

- Managed object comprises network element and management agent
- RMON acts as an agent and a manager
- RMON (Remote Monitoring) probe gathers data from MO, analyses the data, and stores the data
- Communicates the statistics to the manager

Three-Tier Organization Model: Proxy Server

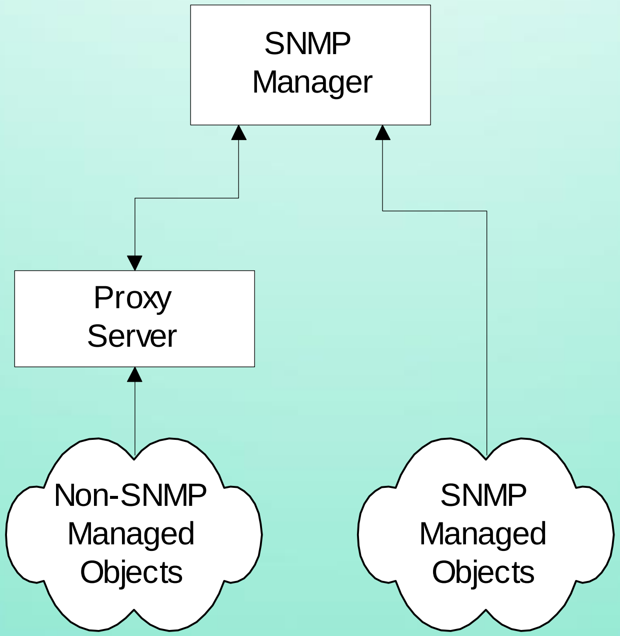


Figure 4.7 Proxy Server Organization Model

Notes

- Proxy server converts non-SNMP data from non-SNMP objects to SNMP compatible objects and messages

Management System Architecture

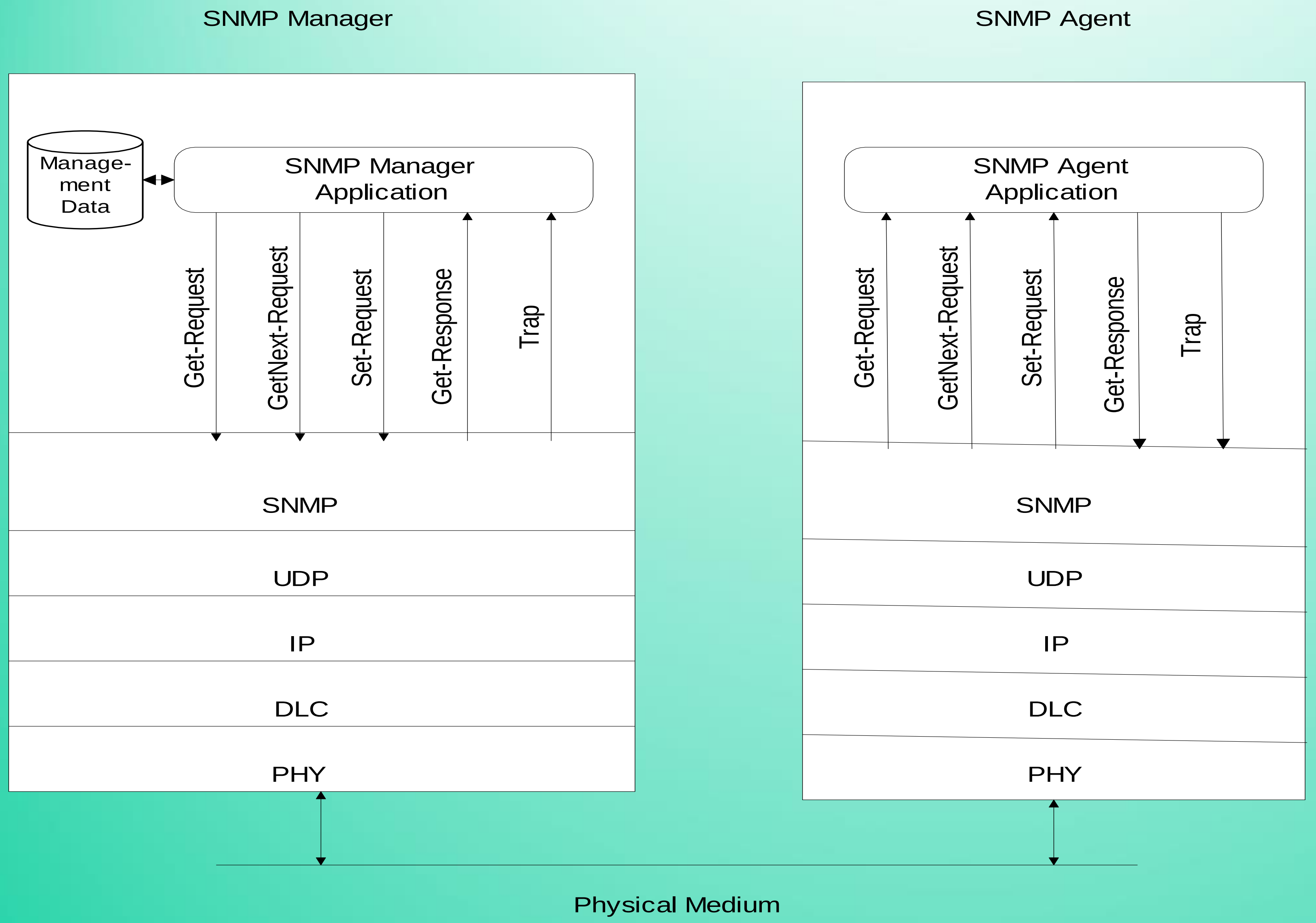


Figure 4.9 SNMP Network Management Architecture

The **Management System Architecture** shows the types of Messages between manager and agent

Notes

- Messages between manager and agent
- Direction of messages - 3 from manager and 2 from agent

SNMP Messages

- Get-Request
    - Sent by manager requesting data from agent
- Get-Next-Request
    - Sent by manager requesting data on the next MO to the one specified
- Set-Request
    - Initializes or changes the value of network element
- Get-Response
    - Agent responds with data for get and set requests from the manager
- Trap
    - Alarm generated by an agent

Notes

Information model

For information to be efficiently exchanged between managers and agents, there has to be common understanding for both syntax and semantics

- Information model deals with:

  - Structure of Management Information (SMI) (RFC 1155)
    - Specifies the structure of management information (Syntax and semantics) using a subset of ASN.1

  - Management Information Base (RFC 1213)
    - Specifies organization of management information in a hierarchical tree-like structure
    - Each object in the MIB (node of the tree) is addressed through an object identifier (OID).

- RFCs can be downloaded from ftp.internic.net/rfc

Management Information Base (MIB)

- Information base contains information about objects
- Organized by grouping of related objects
- Defines relationship between objects
- It is NOT a physical database. It is a *virtual* database that is compiled into management module

Notes

MIB View and Access of an Object

- A managed object has many attributes which compose its management information base
- There are several operations that can be performed on the objects
- A user (manager) can view and perform only certain operations on the object by invoking the management agent
- **The view of the object attributes that the agent perceives is the MIB view**
- The operation that a user can perform is the MIB access

Managed Object

A managed object can be considered to be composed of an object type and an object instance, as shown in Figure 4.10. SMI is concerned only with the object type and not the object instance.

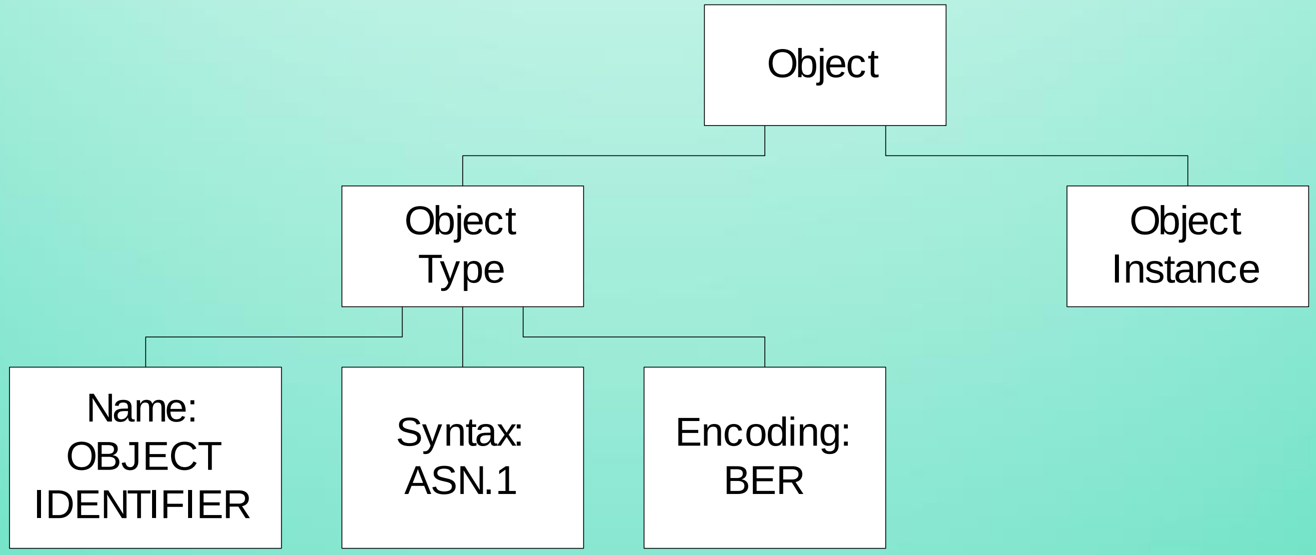


Figure 4.10 Managed Object : Type and Instance

Notes

- Object type and data type are synonymous
- Object identifier is data type, not instance
- Object instance IP address (See Figure 4.2)

Managed Object: Multiple Instances

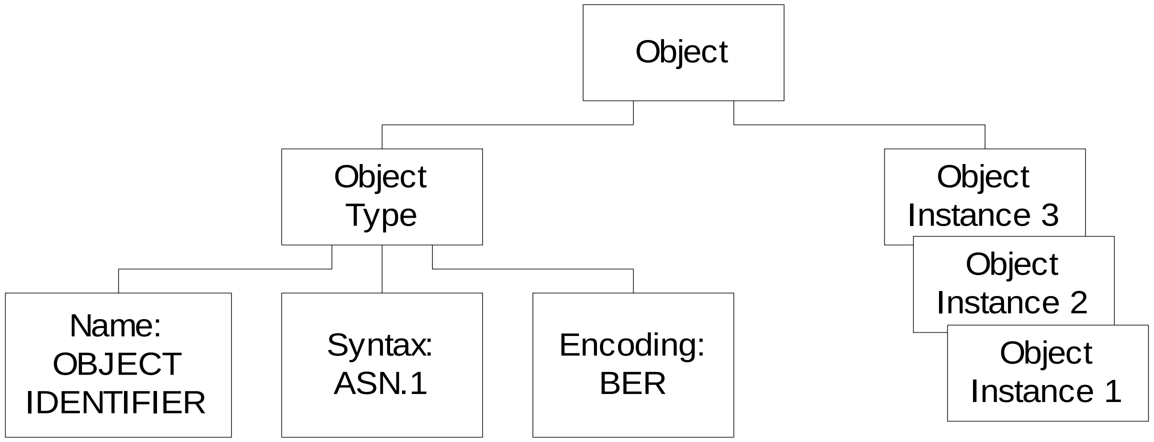


Figure 4.11 Managed Object : Type with Multiple Instances

Notes

- All 3 Com hubs of the same version have identical identifier; they are distinguished by the IP address.
- Each IP address is an instance of the object IP address.
- Basic Encoding Rules

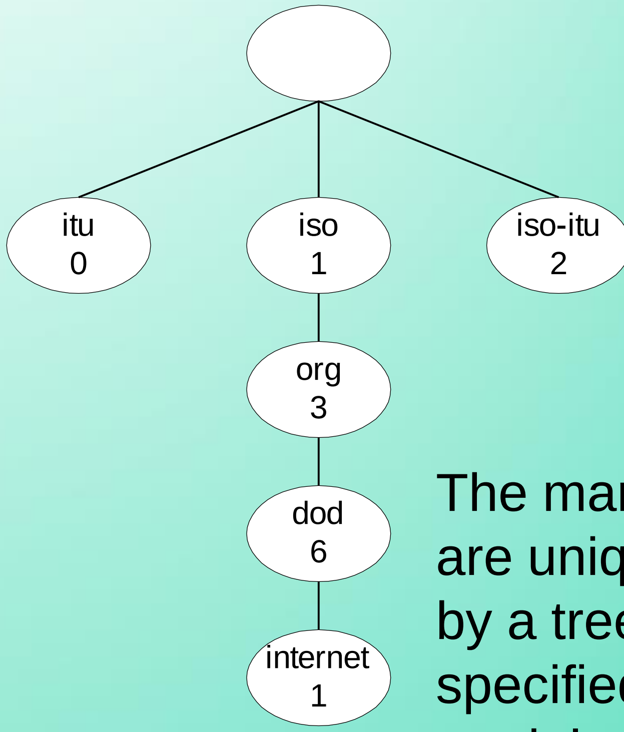
Name

OSI Management Information Tree

Uniquely defined by
- DESCRIPTOR AND
- OBJECT IDENTIFIER

Internet as an organization has an object name and only one instance:

internet OBJECT IDENTIFIER ::=
{iso org(3) dod(6) 1 }.



The managed objects are uniquely defined by a tree structure specified by the OSI model

internet OBJECT IDENTIFIER ::= {iso(1) standard(3) dod(6) internet(1)}

Figure 3.8 **OSI Management Information Tree**

internet OBJECT IDENTIFIER ::= {1 3 6 1}
internet OBJECT IDENTIFIER ::= {iso standard dod internet }
internet OBJECT IDENTIFIER ::= { iso standard dod(6) internet(1) }
internet OBJECT IDENTIFIER ::= { iso(1) standard(3) 6 1 }

Notes

Watch the case

- Example of name: ipAddrTable ip 20

Internet Subnodes

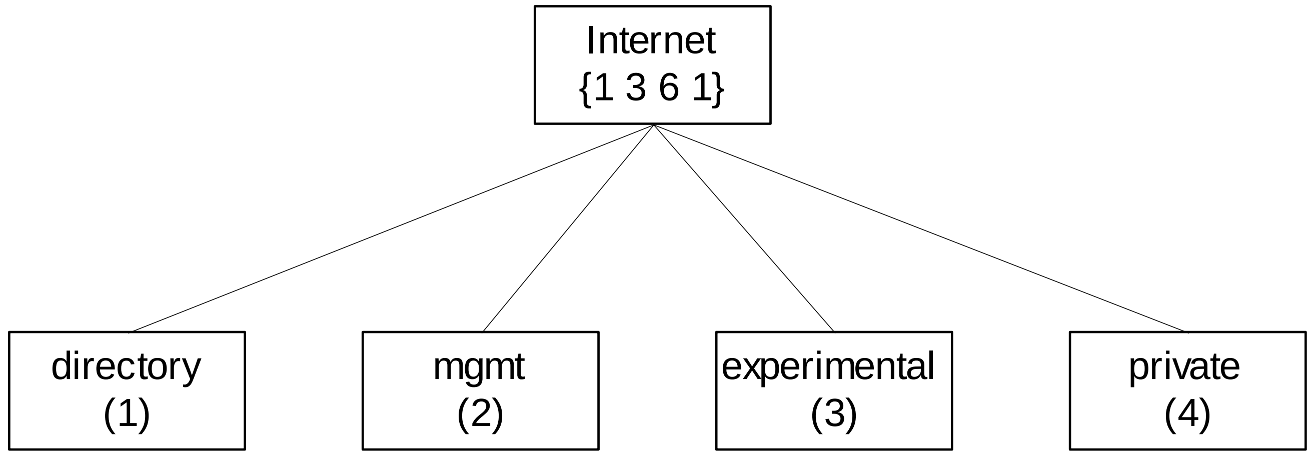


Figure 4.13 Subnodes under Internet Node in SNMPv1

Notes

• directory	OBJECT IDENTIFIER ::= {internet 1}
mgmt	OBJECT IDENTIFIER ::= {internet 2}
experimental	OBJECT IDENTIFIER ::= {internet 3}
private	OBJECT IDENTIFIER ::= {internet 4}

The *experimental*(3) node was created to define objects under IETF experiments. For example, if IANA has approved a number 5 for an experimenter, we would use the OBJECT IDENTIFIER {experimental 5}.

The last node is *private*(4). This is a heavily used node. Commercial vendors can acquire a number under enterprises(1), which is under the private(4) node. Thus, we have

enterprises OBJECT IDENTIFIER ::= {private 1}

or

enterprises OBJECT IDENTIFIER ::= {1 3 6 1 4 1}

Private MIB Example

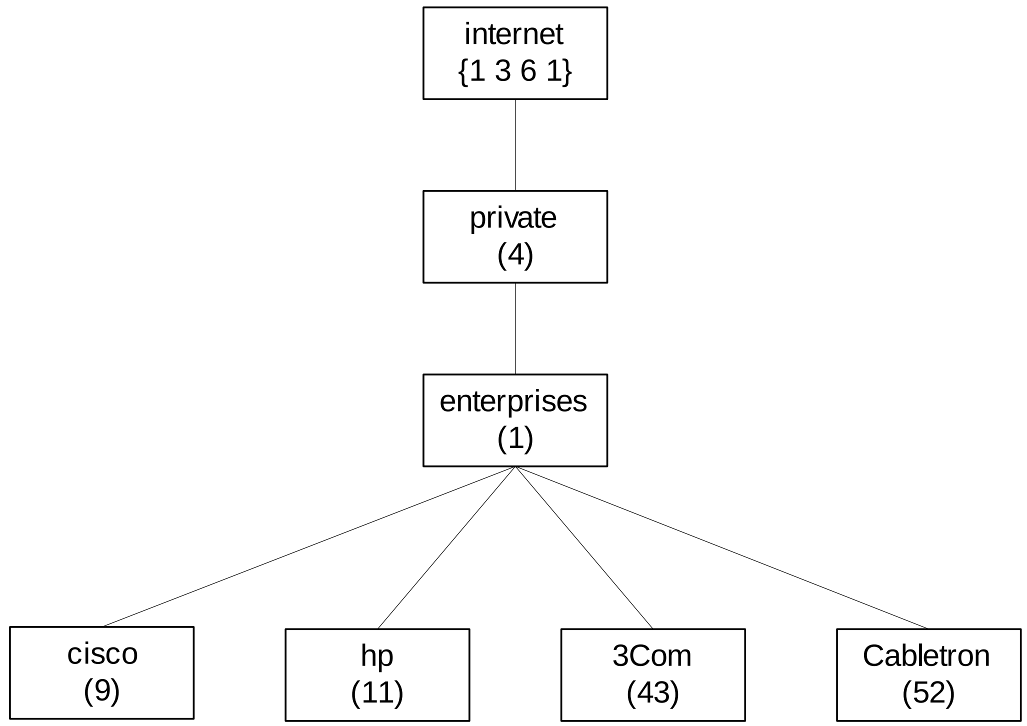


Figure 4.14 Private Subtree for Commercial Vendors

Notes

- *private* MIB intended for vendor equipment
- IANA (Internet Assigned Numbers Authority) assigns identifiers

Figure 4.14 shows an example of four commercial vendors—Cisco, HP, 3Com, and Cabletron who are registered as nodes 9, 11, 43, and 52, respectively, under enterprises(1). Nodes under any of these nodes are entirely left to the discretion of the vendors.

3.6 Abstract Syntax Notation One: ASN.1

- **ASN.1 is more than a syntax; it's a language Addresses both syntax and semantics**

- **Two types of syntax**
  - **Abstract syntax: set of rules that specify data type and structure for information storage**
  - **Transfer syntax: set of rules for communicating information between systems**

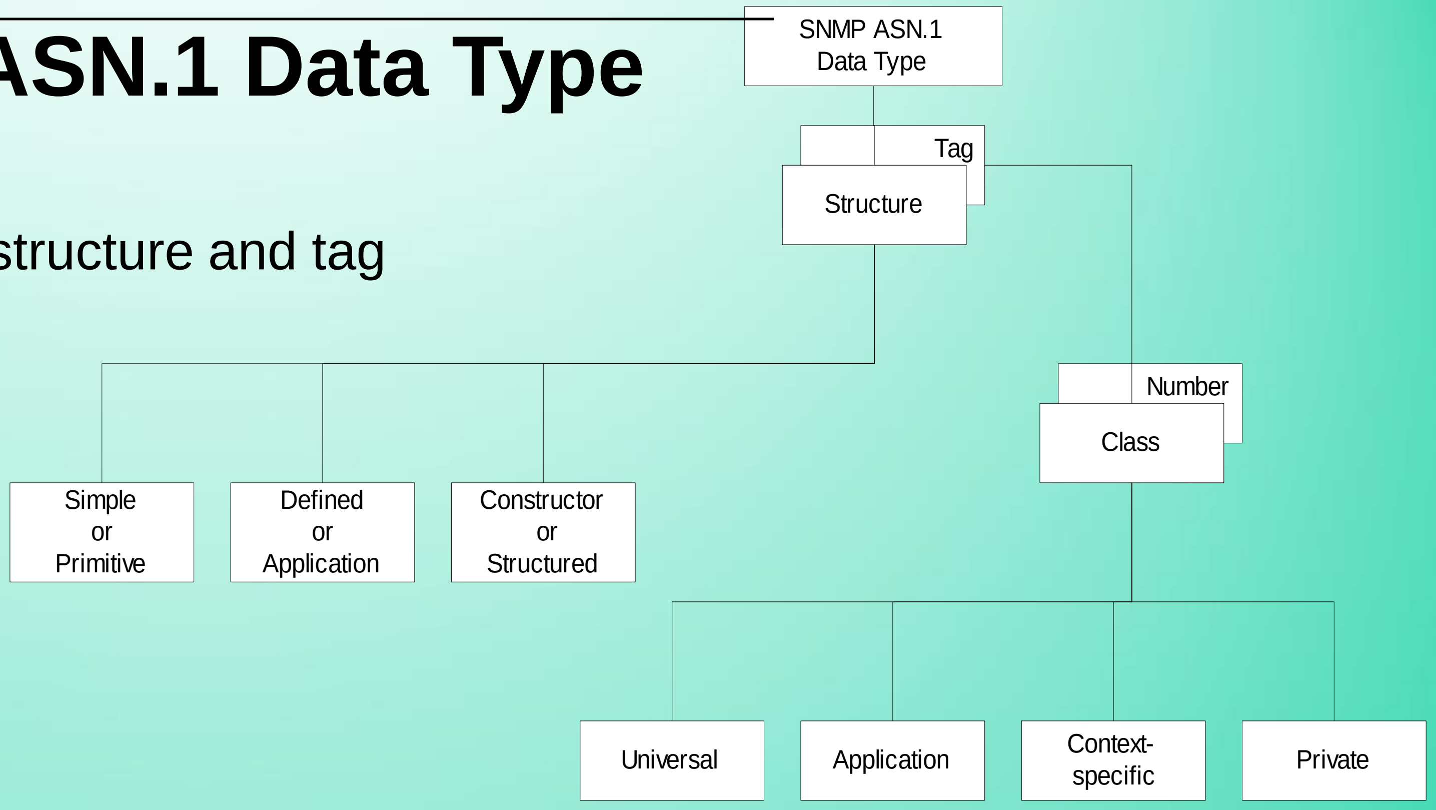
- Makes application layer protocols independent of lower layer protocols

- Can generate machine-readable code: Basic Encoding Rules (BER) is used in management modules

ASN.1 defines the abstract syntax of information but does not restrict the way the information is encoded
ASN.1 facilitates the exchange of structured data especially between application programs over networks by describing data structures in a way that is independent of machine architecture and implementation language.

Notes

SNMP ASN.1 Data Type



In general data types are defined basied on structure and tag

Syntax. ASN.1 syntax that was introduced in Section 3.7 is used to define the structure of object types. Not all constructs of ASN.1 are used in TCP/IP-based SNMP management. Figure 4.15 shows the TCP/IP-based ASN.1 data type. It is very similar to Figure 3.15, but only has three categories under structure.

We observed in Section 3.6.1 that the data type could be either a simple type (also called primitive, atomic, or basic), or it could be structured. In addition, we talked about tag designation, which uniquely identifies the data type irrespective of the syntax version. In general, data types are defined based on structure and tag. The structure is subdivided into four categories. The tag is subdivided into class and tag number. This is shown in Figure 3.15. An object can be uniquely defined by its tag, namely class and tag number. For exchange of information between systems, the structure information is also included.

The four categories of data type structure, shown in Figure 3.15 are, *simple type*, *structured type*, *tagged type*, and *other type*.

Primitive Data Types

Structure	Data Type	Comments
Primitive types	INTEGER	Subtype INTEGER (n1..nN) Special case: Enumerated INTEGER type
	OCTET STRING	8-bit bytes binary and textual data Subtypes can be specified by either range or fixed
	OBJECT IDENTIFIER	Object position in MIB
	NULL	Placeholder

- *get-request* message has NULL for value fields and *get-response* from agent has the values filled in subtype:
  - INTEGER (0..255)
  - OCTET STRING (SIZE 0..255)
  - OCTET STRING (SIZE 8)

Notes

Primitive or simple types are atomic in nature and are: INTEGER, OCTET STRING, OBJECT IDENTIFIER, and NULL. These are also referred to as non-aggregate types.

INTEGER has numerous variations based on the sign, length, range, and enumeration. The reader is referred to Perkins and McGinnis [1997] for a detailed presentation on the subject. When the integer value is restricted by a range, it is called a subtype, as presented in the comments column of Table 4.1, as INTEGER (n1..nN).

The data type ENUMERATED was specified in Section 3.6.2 as a special case of INTEGER data type. In SNMP management, it is specified as INTEGER data type with labeled INTEGER values. The following example of error-status in GetResponse associated with GetRequest-PDU illustrates the use of it. Each enumerated INTEGER has a name associated with it:

Enumerated

- Special case of INTEGER data type
- Example:

```
error-status INTEGER {
        noError(0)
        tooBig(1)
        genErr(5)
        authorizationError(16)
  }
```

Notes

- noError NULL by convention

Any non-zero value indicates the type of error encountered by the agent in responding to a manager's message. As a convention, the value 0 is not permitted in the response message. Thus, a noError message is filled with NULL.

Defined or Application Data Type

Defined types	NetworkAddress	Not used
	IpAddress	Dotted decimal IP address
	Counter	Wrap-around, non-negative integer, monotonically increasing, max 2^32 -1
	Gauge	Capped, non-negative integer, increase or decrease
	TimeTicks	Non-negative integer in hundredths of second units
	Opaque	Application-wide arbitrary ASN.1 syntax, double-wrapped OCTET STRING

Notes

- Defined data types are simple or base types
- Opaque is used to create data types based on previously defined data types

Constructor or Structured Data Type: SEQUENCE

- **List maker** SEQUENCE { <type1>, <type2>,…, <typeN> }

	Object	OBJECT IDENTIFIER	ObjectSyntax
1	ipAdEntAddr	{ipAddrEntry 1}	IpAddress
2	ipAdEntIfIndex	{ipAddrEntry 2}	INTEGER
3	ipAdEntNetMask	{ipAddrEntry 3}	IpAddress
4	ipAdEntBcastAddr	{ipAddrEntry 4}	INTEGER
5	ipAdEntReasmMaxSize	{ipAddrEntry 5}	INTEGER
6	ipAddrEntry	{ipAddrTable 1}	SEQUENCE

If=interface
ipAdEntReasmMaxsize = maximum size for packets reassembling

```
List:   IpAddrEntry  ::=
             SEQUENCE {
                   ipAdEntAddr             IpAddress
                   ipAdEntIfIndex          INTEGER
                   ipAdEntNetMask          IpAddress
                   ipAdEntBcastAddr        INTEGER
                   ipAdEntReasmMaxSize     INTEGER (0..65535)
             }
```

Managed Object IpAddrEntry as a list

The third and last type of structure shown in Figure 4.15 is **constructor** or **structured type**. SEQUENCE and SEQUENCE OF are the only two constructor data types in Table 4.1 that are not base types. They are used to build lists and tables. Note that the constructs SET and SET OF, which are in ASN.1, are not included in the SNMP-based management syntax. SEQUENCE is used to build a list and SEQUENCE OF is used to build a table. We can conceptualize the list as values in a row of a table.

The syntax for list is

SEQUENCE { <type1>, <type2>,…, <typeN> }

where each type is one of ASN.1 primitive types.

The syntax for table is

SEQUENCE OF <entry>

where <entry> is a list constructor.

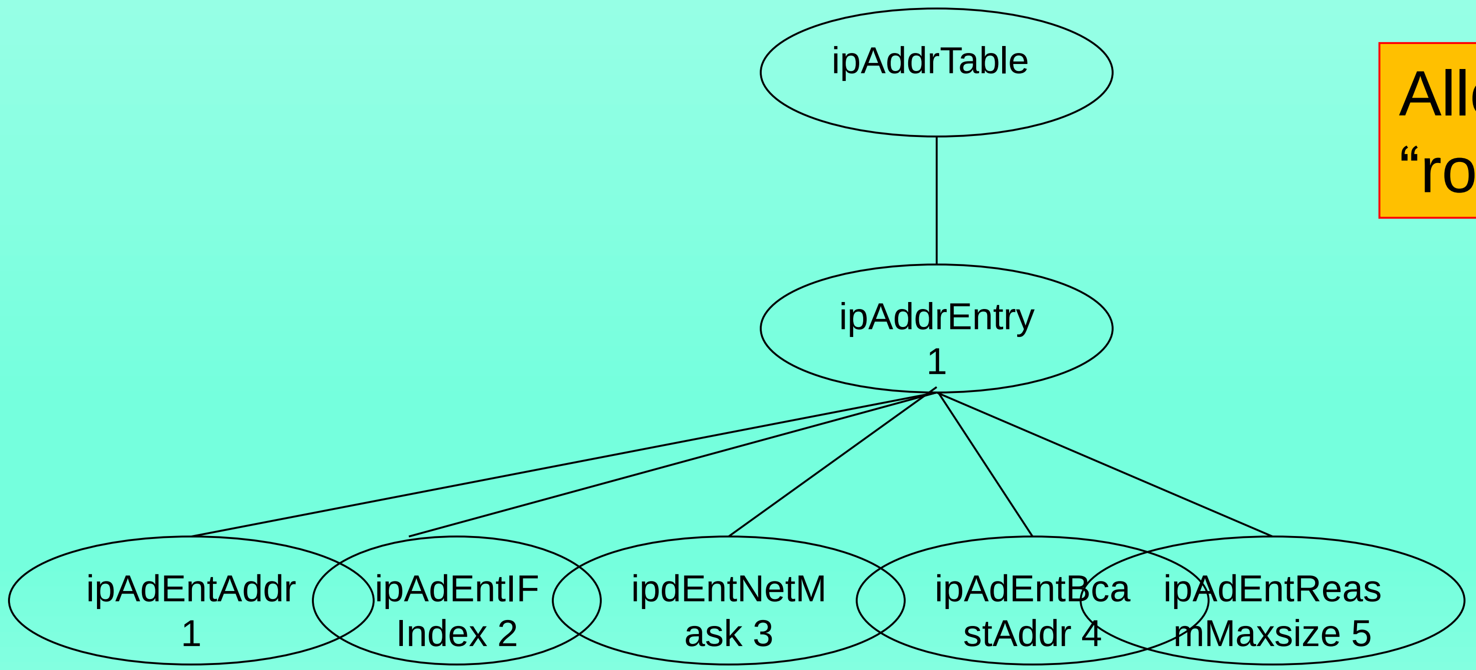
Constructor or Structured Data Type: SEQUENCE OF

SEQUENCE OF <entry> where <entry> is a list constructor

	Object Name	OBJECT IDENTIFIER	Syntax
7	ipAddrTable	{ip 20}	SEQUENCE OF

Table: IpAddrTable ::=
SEQUENCE OF IpAddrEntry

Managed Object ipAddrTable as a table



Allows to create a table = "rows of lists"

SEQUENCE OF Example

Title: System Information : router1.gatech.edu
Name or IP Address: 172.16252.1

Index	Interface	IP address	Network Mask	Network Address	Link Address
23	LEC.1.0	192.168.3.1	255.255.255.0	192.168.3.0	0x00000C3920B4
25	LEC.3.9	192.168.252.15	255.255.255.0	192.168.252.0	0x00000C3920B4
13	Ethernet2/0	172.16..46.1	255.255.255.0	172.16..46.0	0x00000C3920AC
16	Ethernet2/3	172.16.49.1	255.255.255.0	172.16.49.0	0x00000C3920AF
17	Ethernet2/4	172.16.52.1	255.255.255.0	172.16.52.0	0x00000C3920B0
9	Ethernet1/2	172.16.55.1	255.255.255.0	172.16.55.0	0x00000C3920A6
2	Ethernet 0/1	172.16.56.1	255.255.255.0	172.16.56.0	0x00000C39209D
15	Ethernet2/2	172.16.57.1	255.255.255.0	172.16.57.0	0x00000C3920AE
8	Ethernet1/1	172.16.58.1	255.255.255.0	172.16.58.0	0x00000C3920A5
14	Ethernet2/1	172.16.60.1	255.255.255.0	172.16.60.0	0x00000C3920AD

Notes

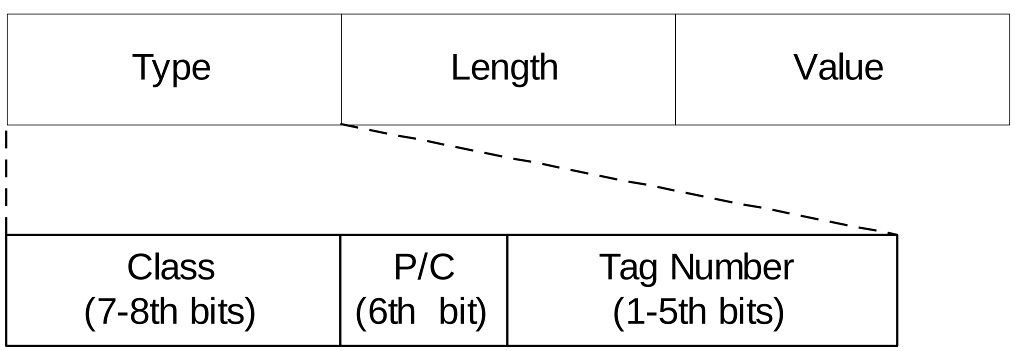
- The above example (Figure 4.3) uses part of the IP MIB discussed for SEQUENCE OF construct.
- Each row of the table is a sequence of (index, interface, Ip address, net mask, net address, link address)

Encoding

In computers, encoding is the process of putting a sequence of characters (letters, numbers, punctuation, and certain symbols) into a specialized format for efficient transmission or storage.

Encoding. SNMPv1 has adopted BER with its TLV for encoding information to be transmitted between agent and manager processes. We covered this in Section 3.8 and illustrated a few ASN.1 data types. SNMP data types and tags are listed in Table 4.3. Encoding rules for various types follow.

- Basic Encoding Rules (BER)
  - Type, Length, and Value (TLV)



Class	8th bit	7th bit
Universal	0	0
Application	0	1
Context-specific	1	0
Private	1	1

P=primitive\c=construct

- SNMP Data Types and Tags

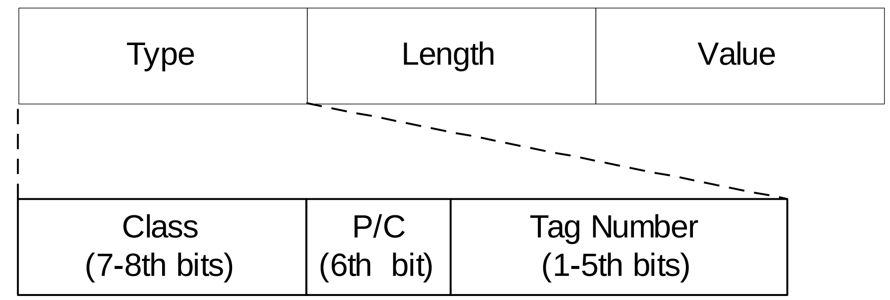
Type	Tag
OBJECT IDENTIFIER	UNIVERSAL 6
SEQUENCE	UNIVERSAL 16
IpAddress	APPLICATION 0
Counter	APPLICATION 1
Gauge	APPLICATION 2
TimeTicks	APPLICATION 3
Opaque	APPLICATION 4

- TLV Type, length, and value are components of the structure

Encoding

https://en.wikipedia.org/wiki/X.690

- Basic Encoding Rules (BER)
  - Type, Length, and Value (TLV)



Within data communication protocols, optional information may be encoded as a type-length-value or TLV element inside a protocol. TLV is also known as tag-length value.
The type and length are fixed in size (typically 1-4 bytes), and the value field is of variable size. These fields are used as follows:

- Type
  A binary code, often simply alphanumeric, which indicates the kind of field that this part of the message represents;
- Length
  The size of the value field (typically in bytes);
- Value
  Variable-sized series of bytes which contains data for this part of the message.

Notes

Managed Object: Structure

We will now specify in detail the SNMP data type format that would serve the basis for defining managed objects.

Structure of Managed Objects. Managed object, as we saw in Section 3.4.2, has five parameters. They are textual name, syntax, definition, access, and status as defined in RFC 1155. For example, *sysDescr* is a data type in the MIB that describes a system. Specifications for the object that describes a system are given in Figure 4.17.

OBJECT:

sysDescr:	{ system 1 }
Syntax:	OCTET STRING
Definition:	"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. It is mandatory that this only contain printable ASCII characters."
Access:	read-only
Status:	mandatory

Figure 4.17 Specifications for System Description

Notes

Managed Object: Macro

Macros for Managed Objects. In order to encode the above information on a managed object to be processed by machines, it has to be defined in a formalized manner. This is done using macros.

```
OBJECT-TYPE MACRO ::=
BEGIN
    TYPE NOTATION ::= "SYNTAX" type(TYPE ObjectSyntax)
        "ACCESS" Access
        "STATUS" Status
    VALUE NOTATION ::= value(VALUE ObjectName)

    Access ::=  "read-only" | "read-write" | "write-only" | "not-accessible"
    Status ::= "mandatory" | "optional" | "obsolete"

END
```

Figure 4.18(a) OBJECT-TYPE Macro [RFC 1155]

```
sysDescr OBJECT-TYPE
    SYNTAX DisplayString (SIZE (0..255))
    ACCESS read-only
    STATUS   mandatory
    DESCRIPTION
        "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. It is mandatory that this only
        contain printable ASCII characters."
::= {system 1 }
```

Figure 4.18(b) Scalar or Single Instance Macro: sysDescr [RFC 1213]

Aggregate Object

- A group of objects
- **Also called tabular objects**
- **Can be represented by a table with**
    - **Columns of objects**
    - **Rows of instances**

A table is a sequence of lists of objects

Table of Objects
↓
List of Objects
↓
Objects

Notes

- Example: IP address table
- table Consists of objects:
    - IP address
    - Interface
    - Subnet mask (which subnet this address belongs to)
    - Broadcast address (value of l.s.b. in IP broadcast address)
    - Largest IP datagram that can be assembled
- Multiple instances of these objects associated with the node

Aggregate M.O. Macro: Table Object

```
ipAddrTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF IpAddrEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
            "The table of addressing
            information relevant to this entity's IP
            addresses."
    ::= {ip 20}
```

```
ipAddrTable   OBJECT-TYPE
        ::= {ip 20}
ipAddrEntry    OBJECT-TYPE
        ::= {ipAddrTable 1}
```

Aggregate M.O. Macro: Entry Object

```
      ipAddrEntry OBJECT-TYPE
          SYNTAX  IpAddrEntry
          ACCESS  not-accessible
          STATUS  mandatory
          DESCRIPTION
                "The addressing information for one of this
entity's IP addresses."

          INDEX   { ipAdEntAddr }
          ::= { ipAddrTable 1 }

        IpAddrEntry ::=
          SEQUENCE {
             ipAdEntAddr
                IpAddress,
             ipAdEntIfIndex
                INTEGER,
             ipAdEntNetMask
                IpAddress,
             ipAdEntBcastAddr
                INTEGER,
             ipAdEntReasmMaxSize
                INTEGER (0..65535)
```

Each entry consists in a list of objects (which will correspond to columns of the table/columnar objects)

Notes

- Index *ipAdEntAddr* uniquely identifies an instance
- May require more than one object in the instance to uniquely identify it

Aggregate M.O. Macro: Columnar Objects

```
ipAdEntAddr OBJECT-TYPE
    SYNTAX  IpAddress
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
            "The IP address to which this entry's
             addressing information pertains."

    ::= { ipAddrEntry 1 }
```

```
ipAdEntReasmMaxSize OBJECT-TYPE
    SYNTAX  INTEGER (0..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
            "The size of the largest IP datagram which this
            entity can re-assemble from incoming IP
            fragmented datagrams received on this interface."
::= { ipAddrEntry 5 }
```

Tabular Representation of Aggregate Object

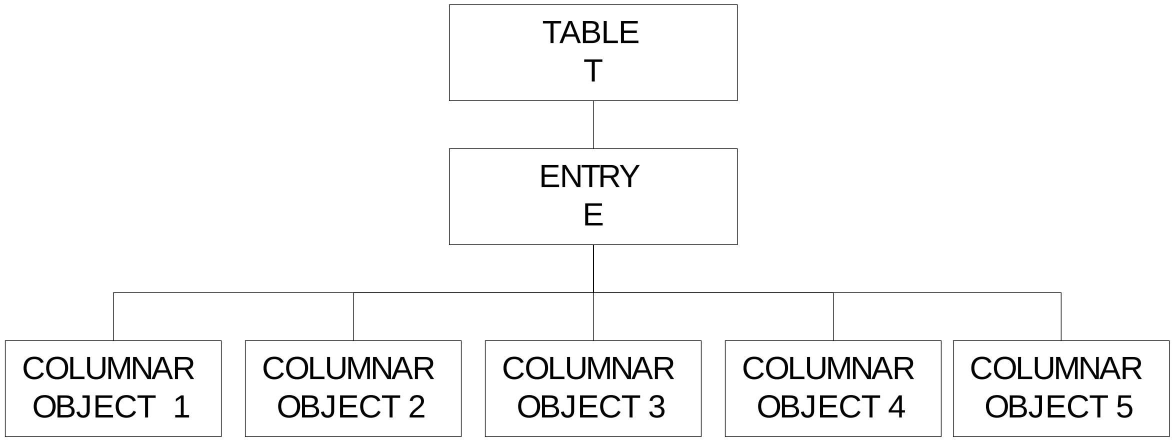


Figure 4.22(a) Multiple Instance Managed Object

Notes

- The objects *TABLE T* and *ENTRY E* are objects that are logical objects. They define the grouping and are not accessible.
- Columnar objects are objects that represent the attributes and hence are accessible.
- Each instance of *E* is a row of columnar objects 1 through 5.
- Multiple instances of *E* are represented by multiple rows.

Tabular Representation of Aggregate Object (cont.)

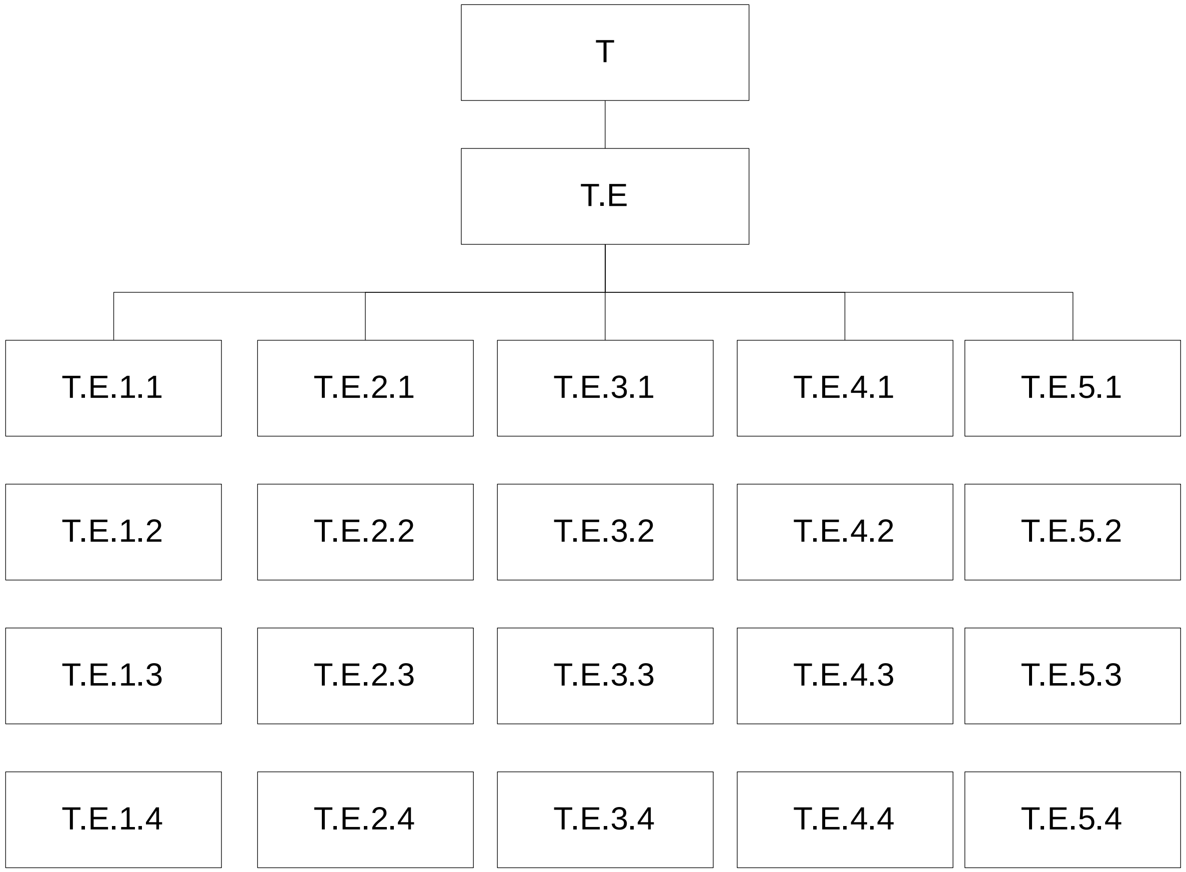


Figure 4.22(b) Example of 5 Columnar Object with 4 Instances (rows)

Notes

- Notice that the column-row numeric designation is reverse of what we are used to as row-column

Multiple Instances of Aggregate Managed Object

```
ipAddrTable {1.3.6.1.2.1.4.20}
        ipAddrEntry (1)
                ipAdEntAddr (1)
                ipAdEntIfIndex (2)
                ipAdEntNetMask (3)
                ipAdEntBcastAddr (4)
                ipAdEntReasmMaxSize (5)

Columnar object ID of  ipAdEntBcastAddr is (1.3.6.1.2.1.4.20.1.4):

iso org dod internet mgmt mib ip ipAddrTable ipAddrEntry ipAdEntBcastAddr
 1   3   6      1       2    1  4     20          1              4
```

Figure 4.23(a) Columnar objects under ipAddrEntry

Row	ipAdEntAddr	ipAdEntIfIndex	IpAdEntNetMask	IpAdEntBcastAddr	IpAdEntReasmMaxSize
1	**123.45.2.1**	1	255.255.255.0	0	12000
2	**123.45.3.4**	3	255.255.0.0	1	12000
3	**165.8.9.25**	2	255.255.255.0	0	10000
4	**9.96.8.138**	4	255.255.255.0	0	15000

Figure 4.23(b) Object instances of ipAddrTable (1.3.6.1.2.1.4.20)

Columnar Object	Row # in (b)	Object Identifier
ipAdEntAddr 1.3.6.1.2.1.4.20.1.1	2	{1.3.6.1.2.1.4.20.1.1.123.45.3.4}
ipAdEntIfIndex 1.3.6.1.2.1.4.20.1.2	3	{1.3.6.1.2.1.4.20.1.2.165.8.9.25}
ipAdEntBcastAddr 1.3.6.1.2.1.4.20.1.4	1	{1.3.6.1.2.1.4.20.1.4.123.45.2.1}
IpAdEntReasmMaxSize 1.3.6.1.2.1.4.20.1.5	4	{1.3.6.1.2.1.4.20.1.5.9.96.8.138}

Figure 4.23(c) Object Id for specific instance

SMI Definition STD 16 / 1155 RFC

The formalized definitions of SMI as presented in STD 16/RFC 1155 is shown here. In addition to the definition of the object type macro, it also specifies the exports of names and object types, as well as the Internet MIB, which is addressed in the next section.

```
RFC1155-SMI DEFINITIONS ::= BEGIN

    EXPORTS -- EVERYTHING
        internet, directory, mgmt, experimental, private, enterprises,
        OBJECT-TYPE, ObjectName, ObjectSyntax, SimpleSyntax,
        ApplicationSyntax, NetworkAddress, IpAddress, Counter, Gauge,
        TimeTicks, Opaque;

    -- the path to the root

    internet      OBJECT IDENTIFIER ::= { iso org(3) dod(6) 1 }

    directory     OBJECT IDENTIFIER ::= { internet 1 }
    mgmt          OBJECT IDENTIFIER ::= { internet 2 }
    experimental  OBJECT IDENTIFIER ::= { internet 3 }
    private       OBJECT IDENTIFIER ::= { internet 4 }

    enterprises   OBJECT IDENTIFIER ::= { private 1 }
```

Notes

- **EXPORTS identifies the objects that any other module could import.**

SMI Definition STD 16 / 1155 RFC (cont.)

```
-- definition of object types

        OBJECT-TYPE MACRO ::=
        BEGIN
          TYPE NOTATION ::= "SYNTAX" type (TYPE ObjectSyntax)
                         "ACCESS" Access
                         "STATUS" Status
          VALUE NOTATION ::= value (VALUE ObjectName)

          Access ::= "read-only" | "read-write" | "write-only" |  "not-accessible"
          Status ::= "mandatory" | "optional" | "obsolete"
        END
```

SMI Definition STD 16 / 1155 RFC (cont.)

```
-- names of objects in the MIB

    ObjectName ::=
        OBJECT IDENTIFIER

    -- syntax of objects in the MIB

    ObjectSyntax ::=
        CHOICE {
            simple
                SimpleSyntax,

            application-wide
                ApplicationSyntax
        }
```

SMI Definition STD 16 / 1155 RFC (cont.)

```
SimpleSyntax ::=
        CHOICE {
            number
                INTEGER,
            string
                OCTET STRING,
            object
                OBJECT IDENTIFIER,
            empty
                NULL
        }
```

```
ApplicationSyntax ::=
        CHOICE {
            address
                NetworkAddress,
            counter
                Counter,
            gauge
                Gauge,
            ticks
                TimeTicks,
            arbitrary
                Opaque

      -- other application-wide types, as they are defined,
         will be added here
        }
```

SMI Definition STD 16 / 1155 RFC (cont.)

```
-- application-wide types

NetworkAddress ::=
   CHOICE {
      internet
         IpAddress
   }
IpAddress ::=
   [APPLICATION 0]        -- in network-byte order
      IMPLICIT OCTET STRING (SIZE (4))
Counter ::=
   [APPLICATION 1]
      IMPLICIT INTEGER (0..4294967295)
Gauge ::=
   [APPLICATION 2]
      IMPLICIT INTEGER (0..4294967295)
TimeTicks ::=
   [APPLICATION 3]
      IMPLICIT INTEGER (0..4294967295)
Opaque ::=
   [APPLICATION 4]        -- arbitrary ASN.1 value,
      IMPLICIT OCTET STRING   --   "double-wrapped"

END
```

Notes

MIB

Let us remember that MIB is a virtual information store (base). Managed objects are accessed via this virtual information base. Objects in the MIB are defined using ASN.1.

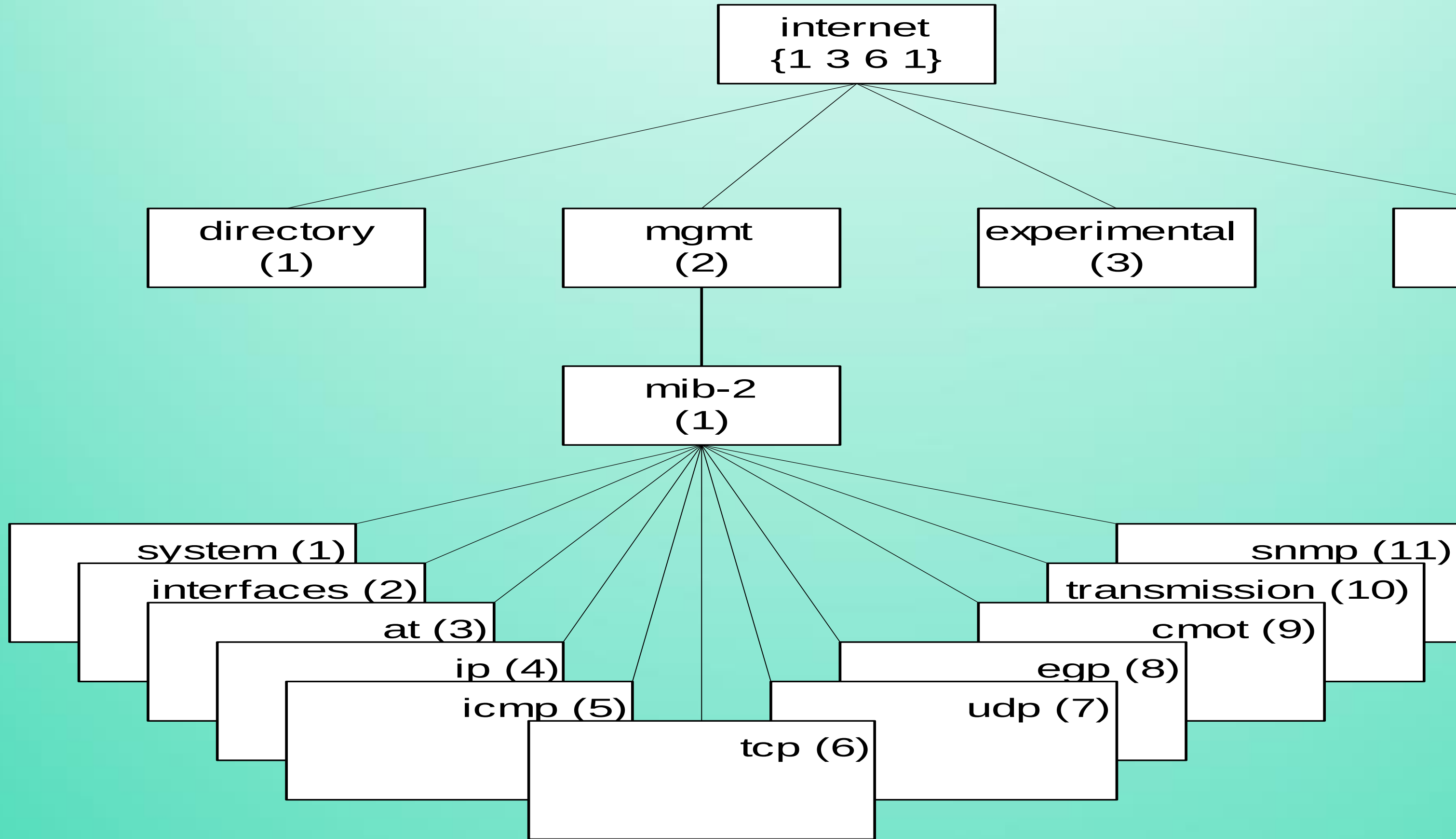


Figure 4.26 Internet MIB-II Group

- MIB-II (RFC 1213) is superset of MIB-I.
- Objects that are related grouped into object groups.
- MIB module comprises module name, imports from other modules, and definitions of current module.
- RFC 1213 **defines eleven groups; expanded later.**

MIB

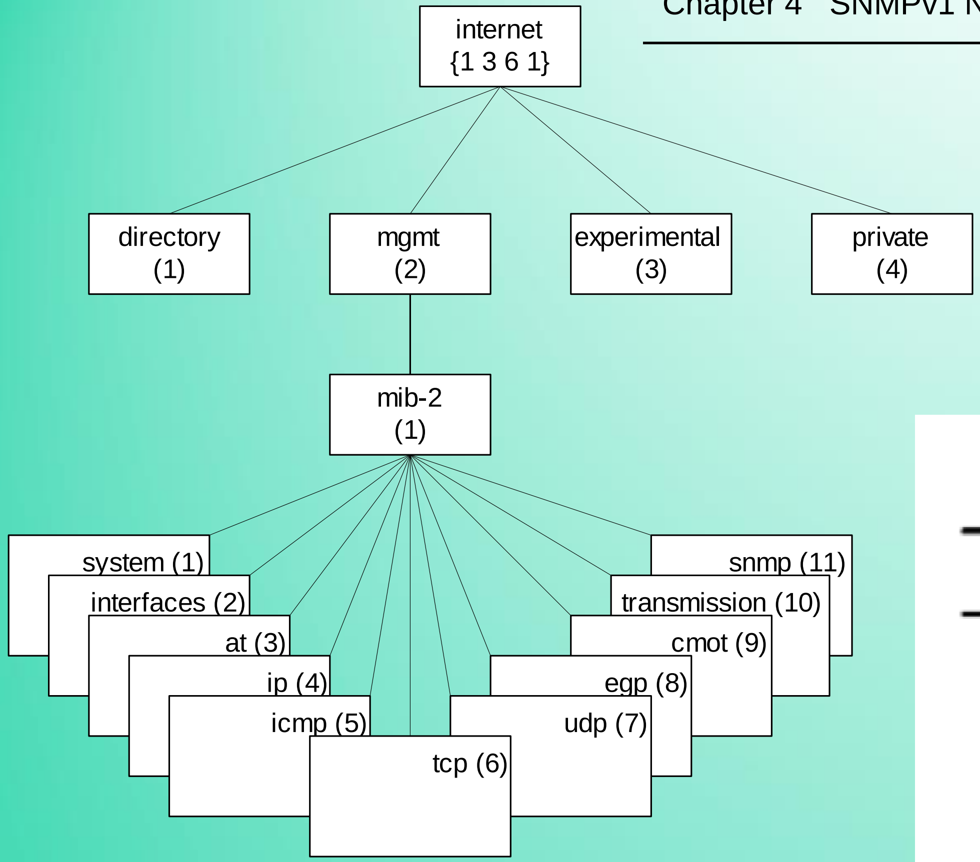


Figure 4.26 Internet MIB-II Group

Table 4.4 MIB-II Groups

GROUP	OID	DESCRIPTION (BRIEF)
system	mib-2 1	System description and administrative information
interfaces	mib-2 2	Interfaces of the entity and associated information
at	mib-2 3	Address translation between IP and physical address
ip	mib-2 4	Information on IP protocol
icmp	mib-2 5	Information on ICMP protocol
tcp	mib-2 6	Information on TCP protocol
udp	mib-2 7	Information on UDP protocol
egp	mib-2 8	Information on EGP protocol
cmot	mib-2 9	Placeholder for OSI protocol
transmission	mib-2 10	Placeholder for transmission information
snmp	mib-2 11	Information on SNMP protocol

System Group

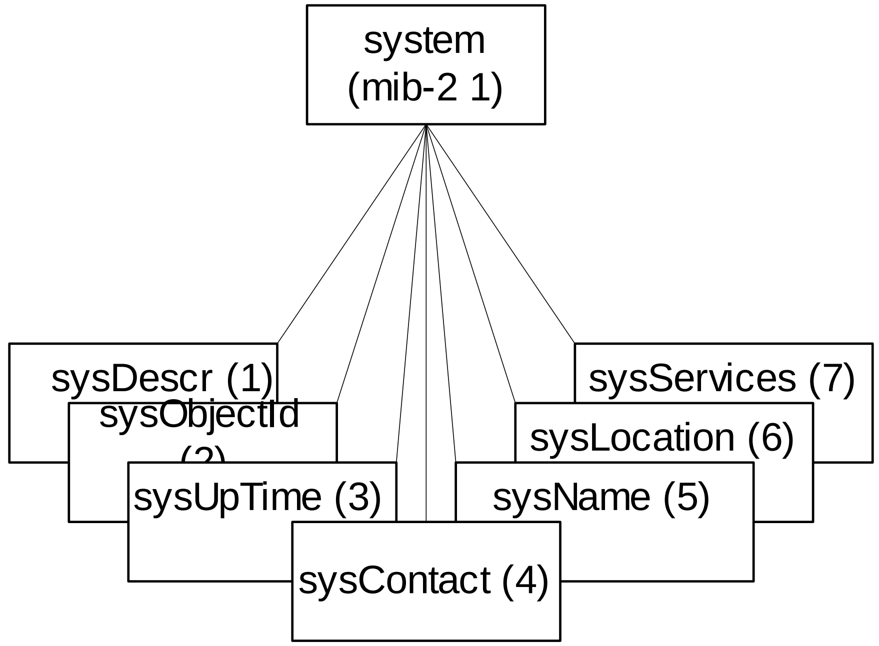


Figure 4.27 System Group

Notes

Entity	OID	Description (brief)
sysDescr	system 1	Textual description
sysObjectID	system 2	OBJECT IDENTIFIER of the entity
sysUpTime	system 3	Time (in hundredths of a second since last reset)
sysContact	system 4	Contact person for the node
sysName	system 5	Administrative name of the system
sysLocation	system 6	Physical location of the node
sysServices	system 7	Value designating the layer services provided by the entity

sysServices

```
sysServices OBJECT-TYPE
        SYNTAX  INTEGER (0..127)
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
             "A value which indicates the set of services that
             this entity primarily 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, 2 raised to (L - 1) is added to the sum.  For
             example, a node which performs primarily routing
             functions would have a value of 4 (2^(3-1)).  In
             contrast, a node which is a host offering
             application services would have a value of 72
             (2^(4-1) + 2^(7-1)).  Note that in the context of
             the Internet suite of protocols, values should be
             calculated accordingly:

                layer functionality
                  1  physical (e.g., repeaters)
                  2  datalink/subnetwork (e.g., bridges)
                  3  internet (e.g., IP gateways)
                  4  end-to-end  (e.g., IP hosts)
                  7  applications (e.g., mail relays)

             For systems including OSI protocols, layers 5 and
             6 may also be counted."
        ::= { system 7 }
```

Notes

Interfaces Group

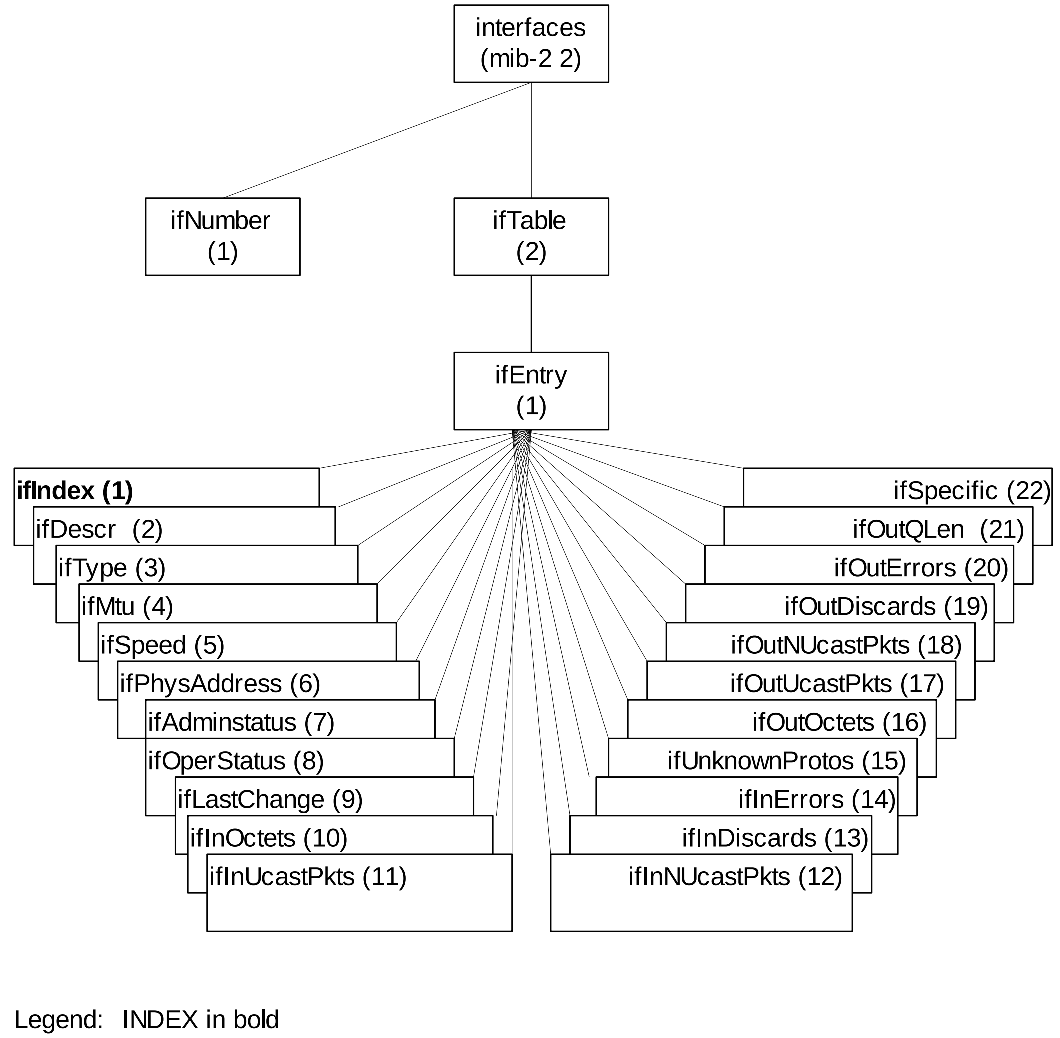
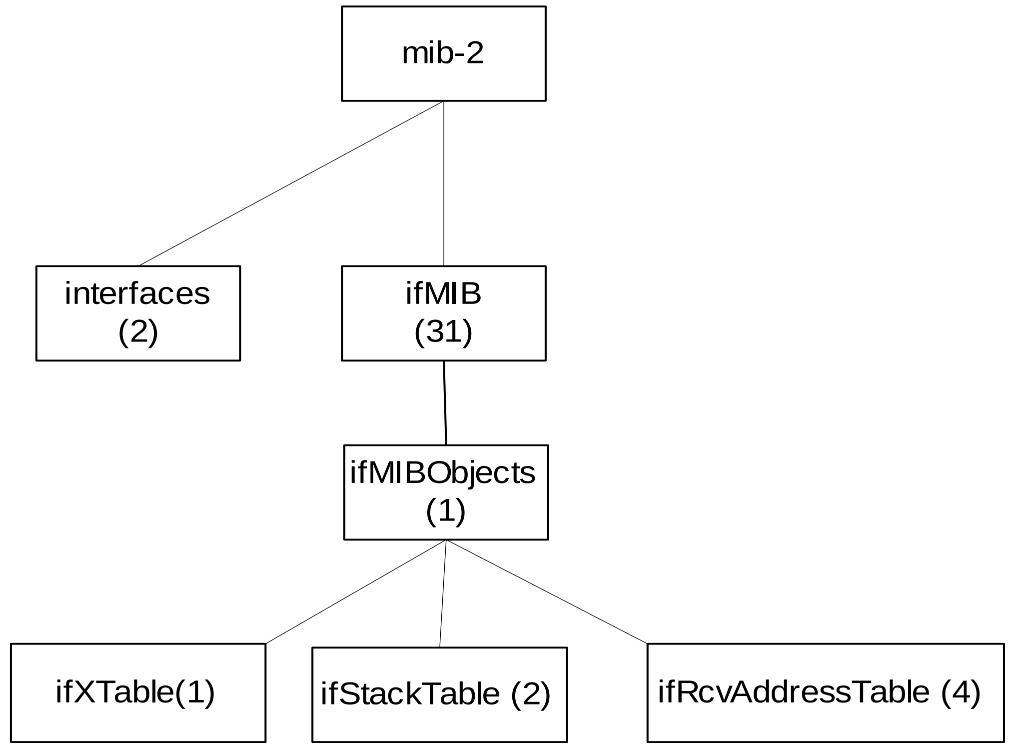


Figure 4.28 Interfaces Group

Notes

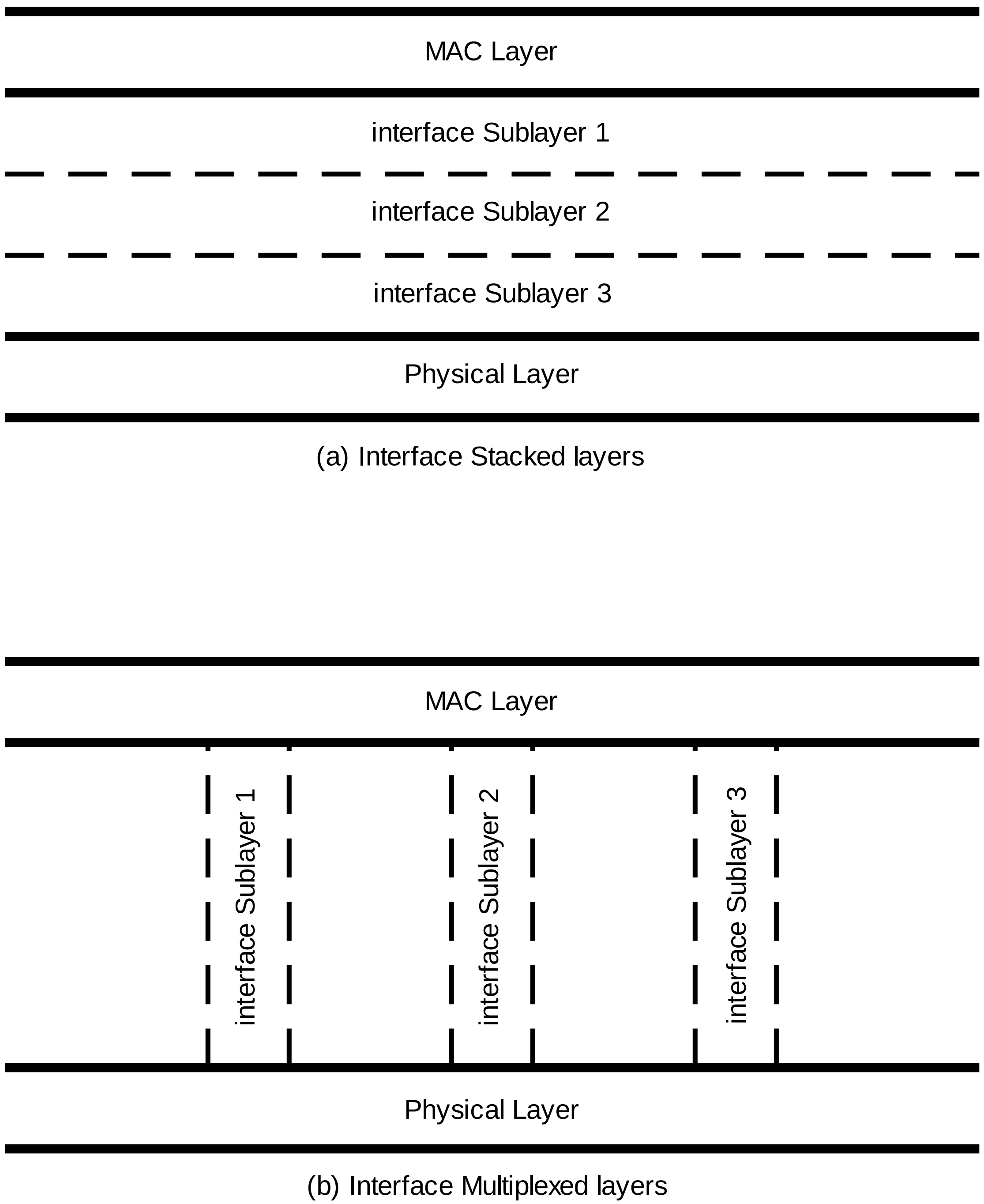
Extension to Interfaces MIB



Notes

- Interfaces MIB limited by maximum number of physical ports
- A physical port may have several conceptual ports
  e.g., channels in cable access network
- *ifMIB* {mib-2 31} created to extend *interfaces MIB*
- *ifMIB* speicifies extension in generic manner
- Specific technology related MIBs supplement details on the conceptual ports
- *ifIndex* in *interfaces* MIB can exceed the maximum number of physical ports
- *ifStack* definition accommodates interface sublayers

Interface Sublayers



DLL can be visualized, in general, as comprising several sublayers. These can either be horizontally stacked or vertically sliced (or "stacked"), as shown in Figures 4.29(a) and (b), respectively. An example of the former is an interface with PPP running over a High data rate Digital Subscriber Line (HDLC) link, which uses an RS232-like connector. An example of the latter is a cable access link with a down-stream channel and several upstream channels.

Figure 4.29 Interface Sublayers

ifEntry

```
IfEntry  OBJECT-TYPE
         SYNTAX     IfEntry
         ACCESS     not-accessible
         STATUS     mandatory
         DESCRIPTION
              "An interface entry containing
              objects at the subnetwork layer and
              below for a particular interface."
         INDEX  {ifIndex}
         ::= {ifTable 1}
```

Notes

- ifEntry specifies the objects in a row in the ifTable.
- Each interface is defined as a row in the table.

ifType

```
ifType OBJECT-TYPE
        SYNTAX  INTEGER {
                other(1),          -- none of the following
                regular1822(2),
                hdh1822(3),
                ddn-x25(4),
                rfc877-x25(5),
                ethernet-csmacd(6),
                iso88023-csmacd(7),
                iso88024-tokenBus(8),
                iso88025-tokenRing(9),
                iso88026-man(10),
                starLan(11),
                proteon-10Mbit(12),
                proteon-80Mbit(13),
                hyperchannel(14),
                fddi(15),
                lapb(16),
                sdlc(17),
                ds1(18),           -- T-1
                e1(19),            -- european equiv. of T-1
                basicISDN(20),
                primaryISDN(21),   -- proprietary serial
                propPointToPointSerial(22),
                ppp(23),
                ……….
```

Notes

- Type of interface below the network layer defined as enumerated integer.

IP Group

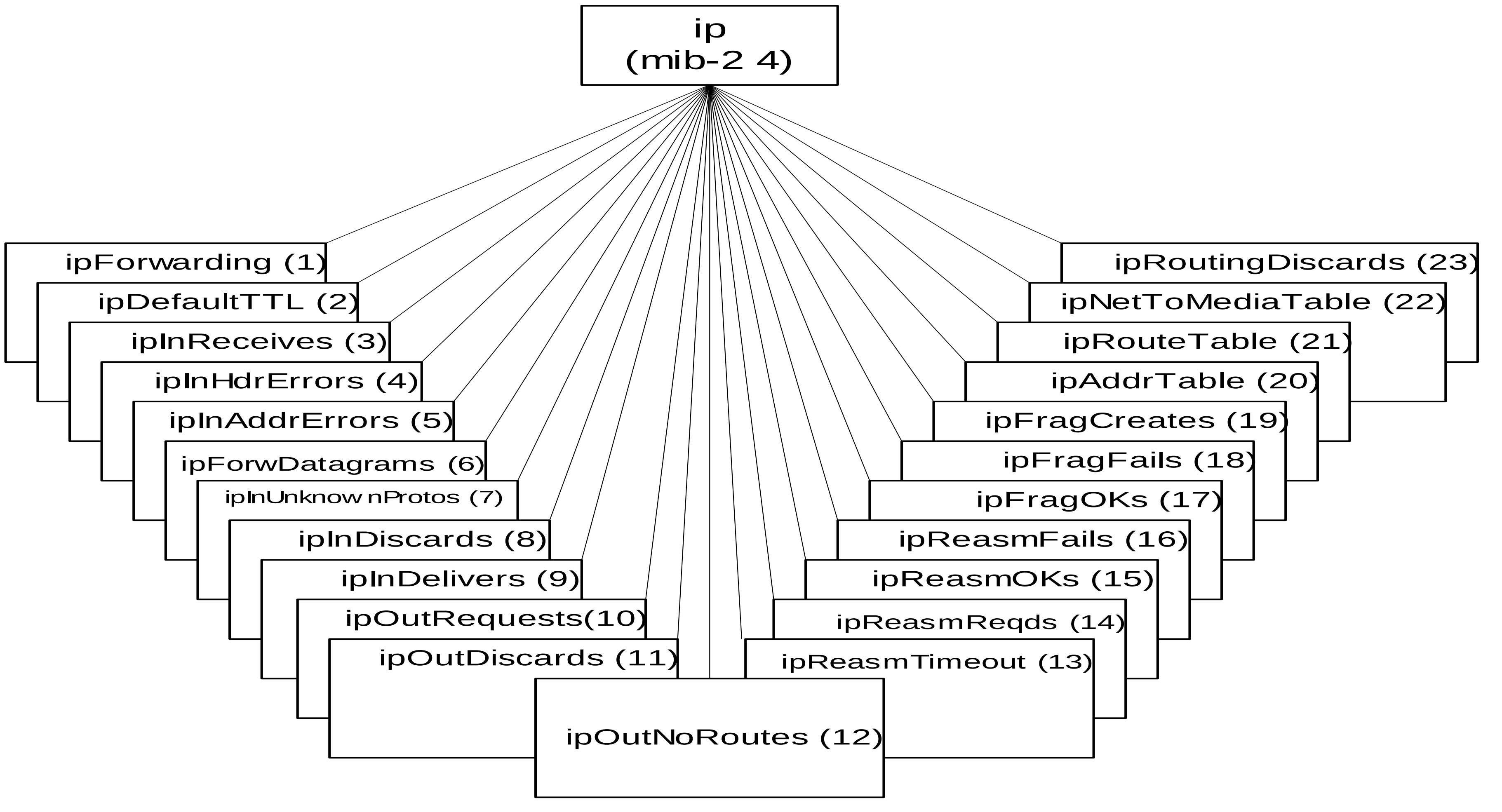


Figure 4.29 IP Group

Notes

- ipForwarding: Gateway(1) and Router(2)
- IP Address Table contains table of IP addresses
- IP Route Table contains an entry for each route
- IP Network-to-Media Table is address translation table mapping IP addresses to physical addresses

IP Address Table

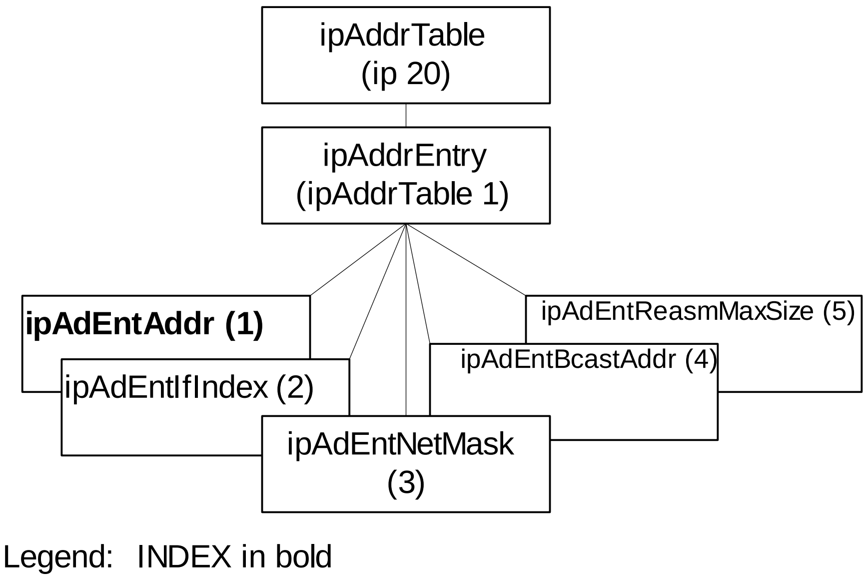


Figure 4.30 IP Address Table

Notes

Entity	OID	Description (brief)
ipAddrTable	ip 20	Table of IP addresses
ipAddrEntry	IpAddrTable 1	One of the entries in the IP address table
ipAdEntAddr	IpAddrEntry 1	The IP address to which this entry's addressing information pertains
ipAdEntIfIndex	IpAddrEntry 2	Index value of the entry, same as ifIndex
ipAdEntNetMask	IpAddrEntry 3	Subnet mask for the IP address of the entry
ipAdEntBcastAddr	IpAddrEntry 4	Broadcast address indicator bit
ipAdEntReasmMaxSize	IpAddrEntry 5	Largest IP datagram that can be reassembled on this interface

IP Routing Table

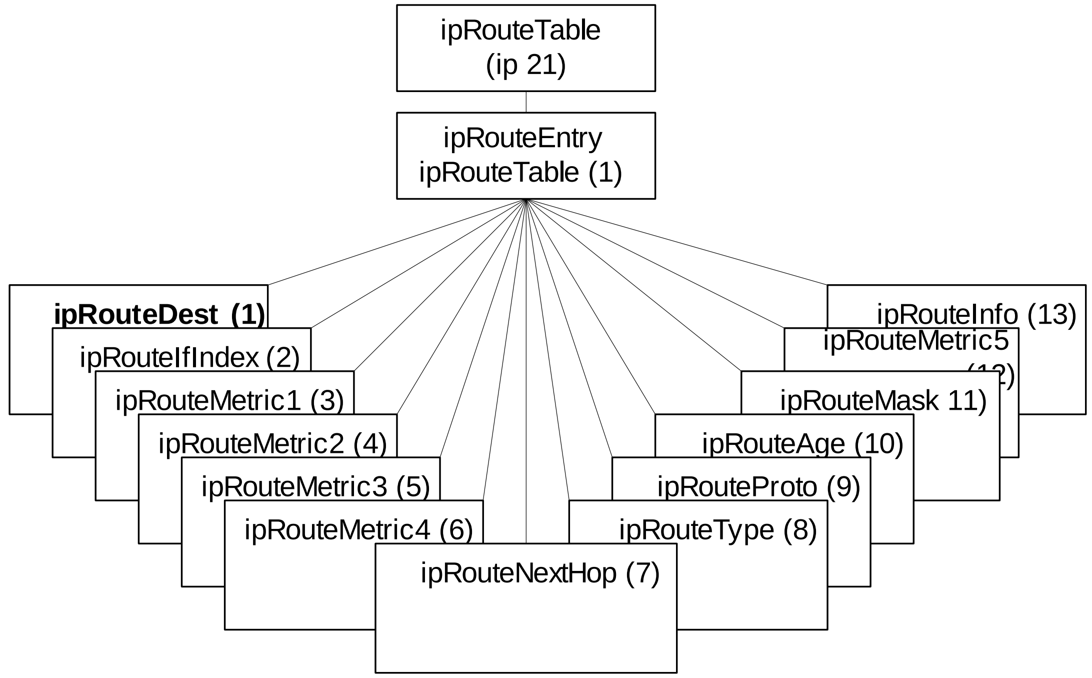


Figure 4.31 IP Routing Table

Entity	OID	Description (brief)
ipRouteTable	ip 21	IP routing table
ipRouteEntry	ipRouteTable 1	Route to a particular destination
ipRouteDest	ipRouteEntry 1	Destination IP address of this route
ipRouteIfIndex	ipRouteEntry 2	Index of interface, same as ifIndex
ipRouteMetric1	ipRouteEntry 3	Primary routing metric for this route
ipRouteMetric2	ipRouteEntry 4	An alternative routing metric for this route
ipRouteMetric3	ipRouteEntry 5	An alternative routing metric for this route
ipRouteMetric4	ipRouteEntry 6	An alternative routing metric for this route
ipRouteNextHop	ipRouteEntry 7	IP address of the next hop
ipRouteType	ipRouteEntry 8	Type of route
ipRouteProto	ipRouteEntry 9	Routing mechanism by which this route was learned
ipRouteAge	ipRouteEntry 10	Number of seconds since routing was last updated
ipRouteMask	ipRouteEntry 11	Mask to be logically ANDed with the destination address before comparing with the ipRouteDest field
ipRouteMetric5	ipRouteEntry 12	An alternative metric for this route
ipRouteInfo	ipRouteEntry 13	Reference to MIB definition specific to the routing protocol

IP Address Translation Table

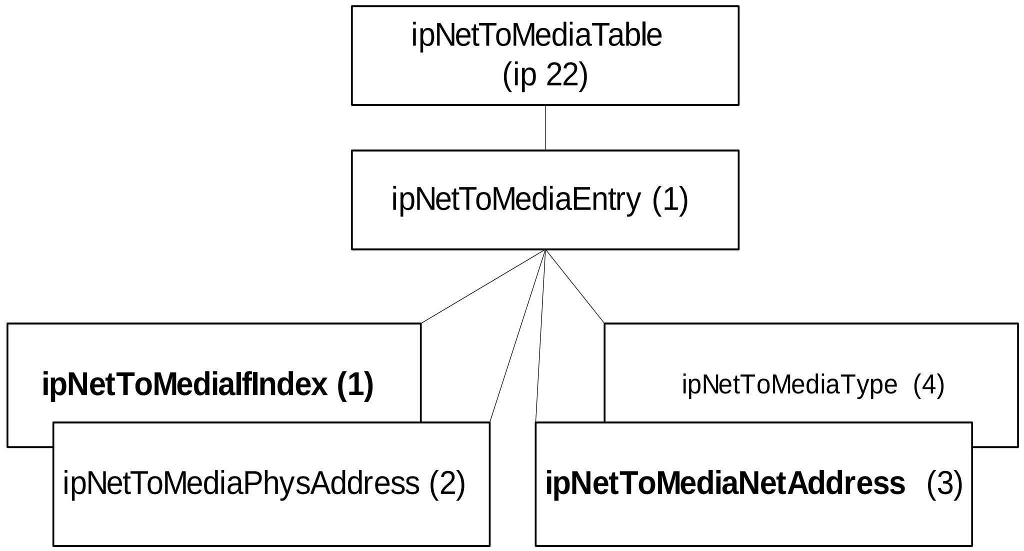


Figure 4.32 IP Address Translation Table

Notes

Entity	OID	Description (brief)
ipNetToMediaTable	ip 22	Table mapping IP addresses to physical addresses
ipNetToMediaEntry	IpNetToMediaTable 1	IP address to physical address for the particular interface
ipNetToMediaIfIndex	IpNetToMediaEntry 1	Interfaces on which this entry's equivalence is effective; same as ifIndex
ipNetToMediaPhysAddress	IpNetToMediaEntry 2	Media dependent physical address
ipNetToMediaNetAddress	IpNetToMediaEntry 3	IP address
ipNetToMediaType	IpNetToMediaEntry 4	Type of mapping

ICMP Group

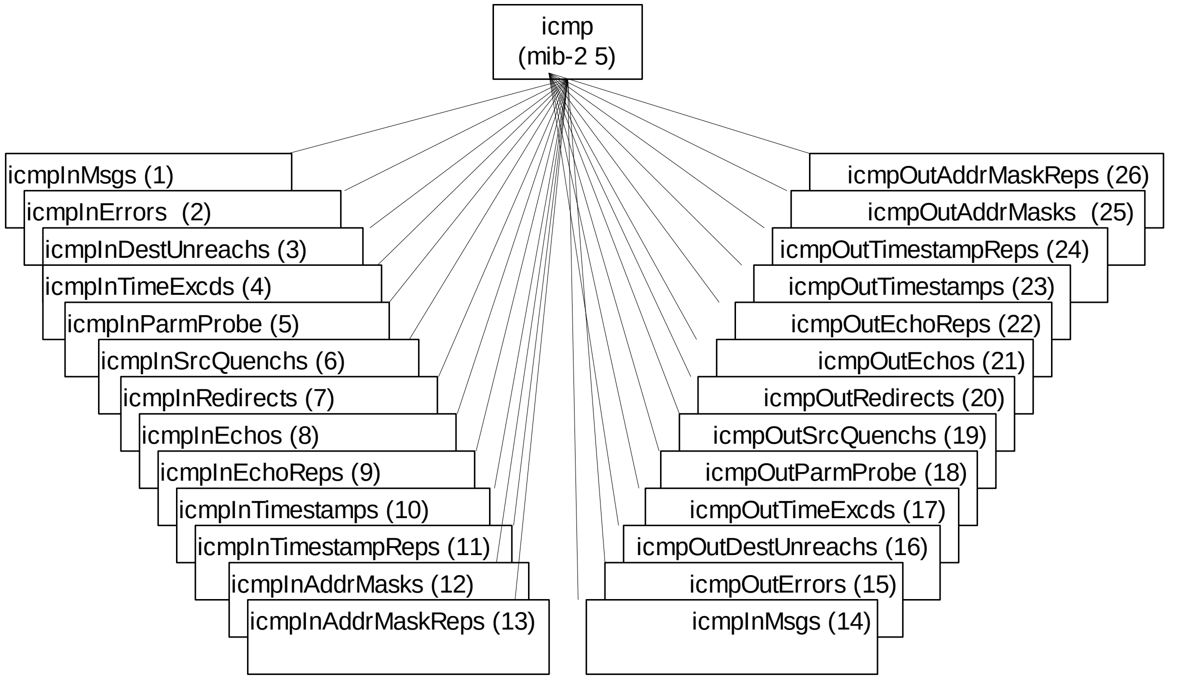


Figure 4.34 ICMP Group

Notes

- Objects associated with *ping*
  - icmpOutEchos # ICMP echo messages sent
  - icmpInEchoReps # ICMP echo reply messages received
- Objects associated with *traceroute/tracert*
  - icmpInTimeExcs # ICMP time exceeded messages received

TCP Group

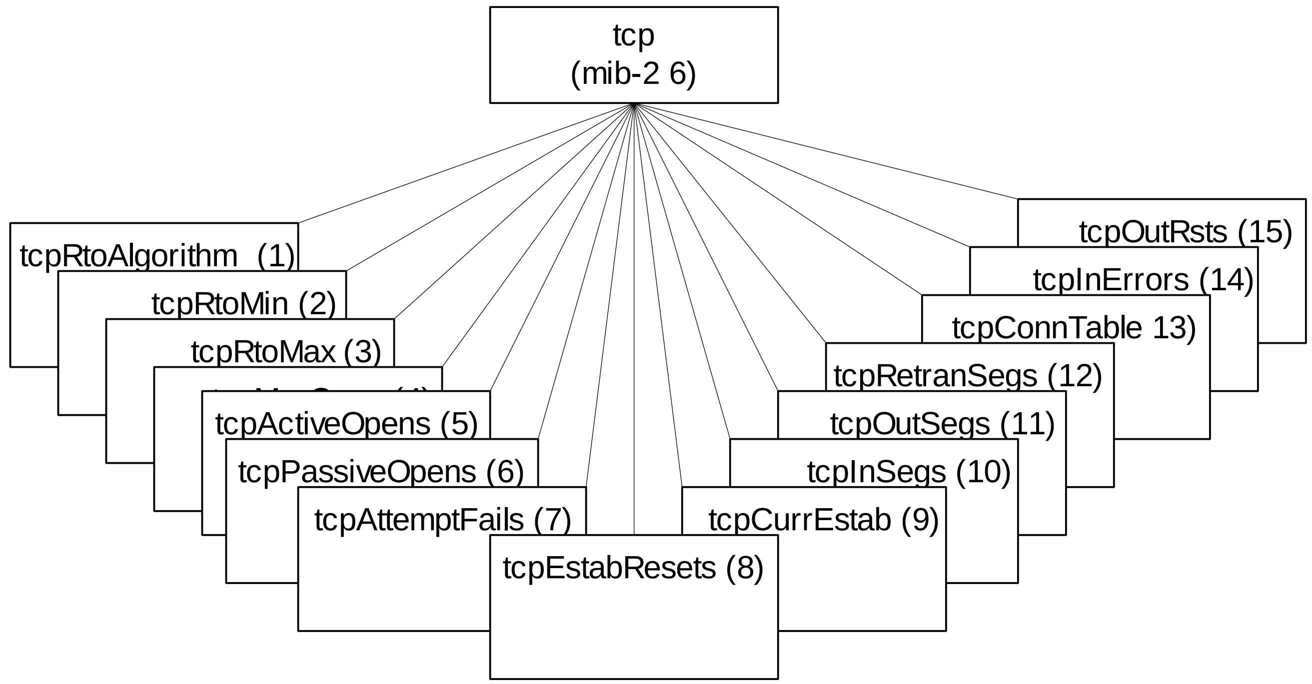


Figure 4.35 TCP Group

Notes

- Connection-oriented transport protocol group
- Has one table

TCP Connection Table

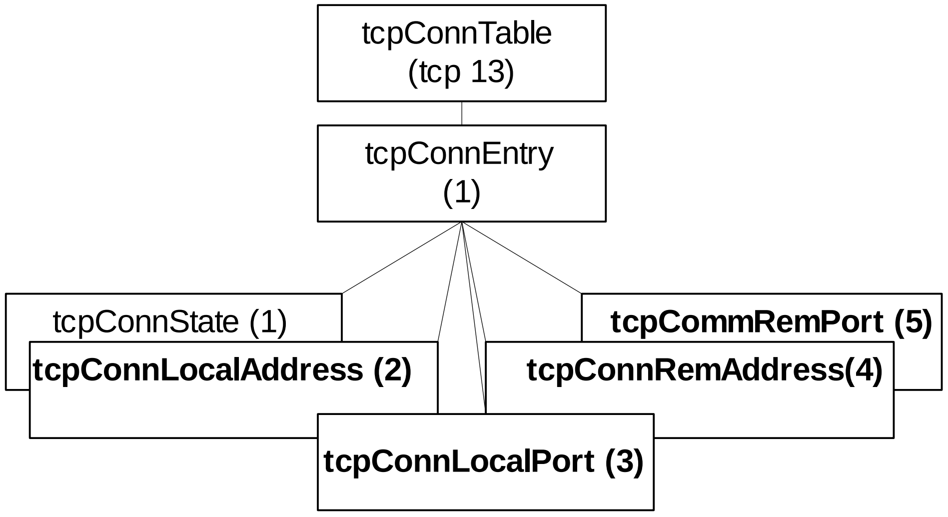


Figure 4.36 TCP Connection Table

Notes

Entity	OID	Description (brief)
tcpConnTable	tcp 13	TCO connection table
tcpconnEntry	TcpConnTable 1	Information about a particular TCP connection
tcpConnState	TcpConnEntry 1	State of the TCP connection
tcpConnLocalAddress	TcpConnEntry 2	Local IP address
tcpConnLocalPort	TcpConnEntry 3	Local port number
tcpConnRemAddress	TcpConnEntry 4	Remote IP address
tcpConnRemPort	TcpConnEntry 5	Remote port number

UDP Group

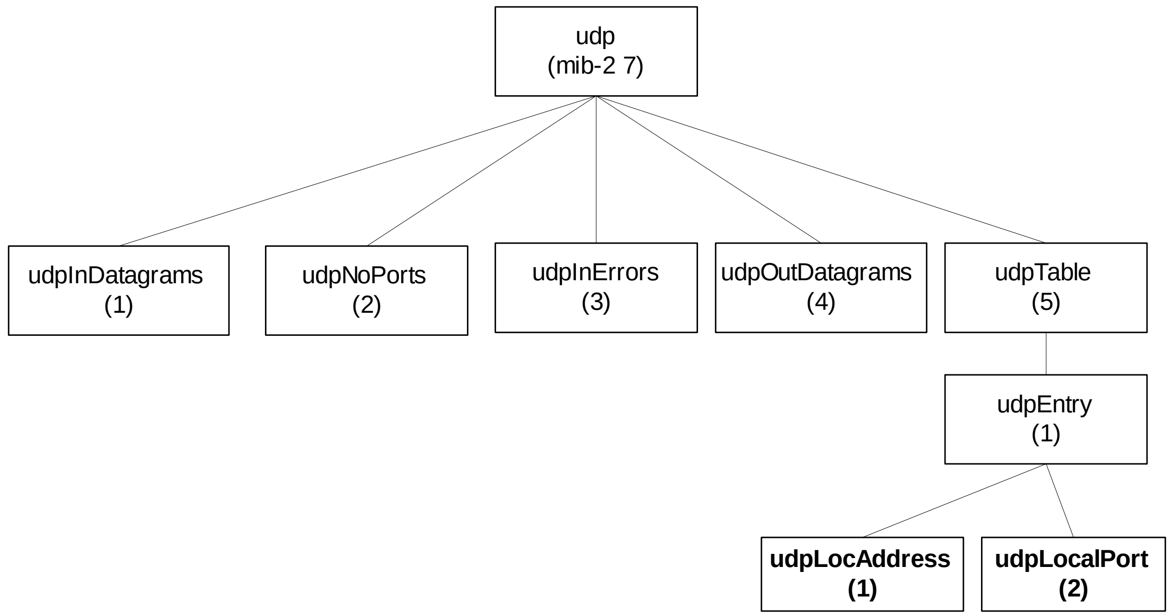


Figure 4.37 UDP Group

Notes

- Connectionless transport protocol group
- Has one table, UDP table

Entity	OID	Description (brief)
udpInDatagrams	udp 1	Total number of datagrams delivered to the users
udpNoPorts	udp 2	Total number of received datagrams for which there is no application
udpInErrors	udp 3	Number of received datagrams with errors
udpOutDatagrams	udp 4	Total number of datagrams sent
udpTable	udp 5	UDP Listener table
udpEntry	udpTable 1	Information about a particular connection or UDP listener
udpLocalAddress	udpEntry 1	Local IP address
udpLocalPort	udpEntry 2	Local UDP port