Exercice 1 Questions théoriques

On s'intéresse à la MIB SNMP.

1. Quel est l'intérêt d'avoir un arbre de référence qui est à la fois unique et « extensible » ?
2. Le type *ASN.1* de la valeur d'un objet géré a-t-il un rapport avec la référence de l'objet ?
3. Arbres et MIB (en utilisant les arbres fournis en annexe et dans le cours).
   a) Donner le nom de l'objet correspondant à l'OID suivant 1.3.6.1.2.1.4.22. A quoi correspond cet objet ?

Exercice 2 : Analyse d'un fichier de MIB

Voir annexe1 : définition de la MIB-2.

1. Quels sont les éléments (attributs) qui définissent un objet géré ?
2. Quelle est la signification selon vous de : `mib-2   OBJECT IDENTIFIER ::= {mgmt 1 }?`
3. Quelle est la valeur par rapport à la MIB-2 de l'OID IP ? Par rapport à MGMT ?
4. Quelle est la valeur de l'OID de *ifTable* ?
5. Que signifie **mandatory** ?
6. Que représente la suite : IfEntry :: = SEQUENCE {.... ? Donner une représentation de la table interface. Quel index est utilisé par cette table ?
7. Donner l'OID de l'IfMtu de la 3ème interface de la machine en partant de l'objet **interfaces**.

Exercice 3

Dans cet exercice on utilise l'annexe 2.

1. Quel index est utilisé pour parcourir la table de routage ? Quel OID doit-on demander pour obtenir le prochain saut associé à la route par défaut ? Quel OID doit-on demander pour obtenir le masque associé au réseau de destination 192.168.1.0/24 ?
2. Donner la table de routage du poste qui a été sondé via SNMP.

Annexe1

```
Version raccourcie et simplifiée de la MIB-2
RFC1213-MIB DEFINITIONS ::= BEGIN
IMPORTS
         mgmt, NetworkAddress, IpAddress, Counter, Gauge, TimeTicks
                  FROM RFC1155-SMI
         OBJECT-TYPE
                  FROM RFC 1212;

         mib-2    OBJECT IDENTIFIER ::= {mgmt 1 }

-- groups in MIB-II

system OBJECT IDENTIFIER ::= { mib-2 1 }
interfaces OBJECT IDENTIFIER ::= { mib-2 2 }
at OBJECT IDENTIFIER ::= { mib-2 3 }
ip OBJECT IDENTIFIER ::= {mib-2 4 }
icmp OBJECT IDENTIFIER ::= { mib-2 5 }
...

-- the Interfaces table

-- The Interfaces table contains information on the entity's interfaces. Each
-- interface is thought of as being attached to a 'subnetwork.' Note that this
-- terni should not be confused with 'subnet,' which refers to an addressing
-- partitioning scheme used in the Internet suite of protocols.

ifTable OBJECT-TYPE
```

```
          SYNTAX SEQUENCE OF IfEntry
          ACCESS not-accessible
          STATUS mandatory
          DESCRIPTION
                    "A list of interface entries. The number of entries is given
                    by the value of ifNumber."
          ::={ interfaces 2 }

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

IfEntry ::=
SEQUENCE {
          ifIndex
                    INTEGER,
          ifDescr
                    DisplayString,
          ifType
                    INTEGER,
          ifMtu
                    INTEGER,
          ifSpeed
                    Gauge,
          ifPhysAddress
                    PhysAddress,
          ifAdminStatus
                    INTEGER,
          ifOperStatus
                    INTEGER,
          ifLastChange
                    TimeTicks,
          ifInOctets
                    Counter,
          ifInUcastPkts
                    Counter,
          ifInNUcastPkts
          Counter,
...
          ifSpecific
                    OBJECT IDENTIFIER
}
ifIndex OBJECT-TYPE
          SYNTAX INTEGER
          ACCESS read-only
          STATUS mandatory
          DESCRIPTION
          "A unique value for each interface. Its value ranges between 1 and the
value of ifNumber. The value for each interface must reinain constant at least
from one reinitialization of the entity's network-management system to the next
reinitialization."

          ::= { ifEntry 1 }
ifDescr OBJECT-TYPE
          SYNTAX DisplayString (SIZE (0..255))
```

```
         ACCESS read-only
         STATUS mandatory
         DESCRIPTION
         "A textual string containing information about the interface. This
         string should include the naine of the manufacturer, the product
         naine, and the version of the hardware interface."
::={ ifEntry 2 }
```

MIB 2	System (1)				
	Interface (2)	Ifnumber (1)			

		IfTable(2)	IfEntry (1)	ifIndex(1)
				ifDescr(2)
				ifType(3)
				ifMtu(4)
				ifSpeed(5)
				ifPhys@(6)
				ifOperstatus(7)
				ifAdminstatus(8)
				ifLastchange(9)
				IfInoctets (10)
				....	

	IP (4)	ipforwarding(1)
		ipdefaultTTL(2)
		ipInReceives(3)
		ipInHdrErrors(4)
		ipInAddrErrors (5)
		ipForwDatagrams (6)
		ipInUnknownProtos (7)
		ipInDiscards(8)			

		....			
		ipFragOKs (17) IpFragFails (18) ipFragCreates (19)			
		ipAddrTable (20)	Ipaddrentry(1)	IpAdEntAddr (1) IpAdEntIfIndex (2) IpAdEntNetMask (3) ipAdEntBcastAddr (4) IpAdEntReasmMaxSize (5)	
		IpRoutingTable (21)	iprouteEntry(1)	IpRouteDest (1) IpRouteIfIndex (2) ipRouteMetric1(3) ipRouteMetric2 (4) ipRouteMetric3 (5) ipRouteMetric4 (6) ipRouteNextHop (7)	

				ipRouteType (8)	Other(1)
					.....
					Remote(4)

				ipRouteProto (9)	other(1)
					Local(2)
					Netmgmt(3)
					Icmp(4)
					..
					BGP(14)

				ipRouteAge (10) ipRouteMask (11)	
		IpNetTomediaTable (22)	IpNettomediaentry (1)	IpNetToMediaifIndex (1) IpNetToMediaPhys@ IPNetToMediaNet@ IPNetToMediaType	

Annexe2

Capture d’échanges SNMP: à gauche objet demandé/ à droite valeur.

MIB2.4.20.1.1.127.0.0.1	127.0.0.1		MIB2.4.22.1.2.2.192.168.136.1	
MIB2.4.20.1.1.192.168.136.135	192.168.136.135		MIB2.4.22.1.3.2.192.168.136.1	192.168.136.1
MIB2.4.20.1.2.127.0.0.1	1		MIB2.4.22.1.4.2.192.168.136.1	3
MIB2.4.20.1.2.192.168.136.135	2			
MIB2.4.20.1.3.127.0.0.1	255.0.0.0		.1.3.6.1.2.1.2.2.1.1.5	5
MIB2.4.20.1.3.192.168.136.135	255.255.255.0		.1.3.6.1.2.1.2.2.1.2.1	lo
MIB2.4.20.1.4.127.0.0.1	0		.1.3.6.1.2.1.2.2.1.2.2	eth0
MIB2.4.20.1.4.192.168.136.135	1		.1.3.6.1.2.1.2.2.1.2.3	eth1
MIB2.4.21.1.1.0.0.0.0	0.0.0.0		.1.3.6.1.2.1.2.2.1.2.4	eth2
MIB2.4.21.1.1.169.254.0.0	169.254.0.0		.1.3.6.1.2.1.2.2.1.2.5	sit0
MIB2.4.21.1.1.192.168.136.0	192.168.136.0		.1.3.6.1.2.1.2.2.1.3.1	24
MIB2.4.21.1.2.0.0.0.0	2		.1.3.6.1.2.1.2.2.1.3.2	6
MIB2.4.21.1.2.169.254.0.0	2		.1.3.6.1.2.1.2.2.1.3.3	6
MIB2.4.21.1.2.192.168.136.0	2		.1.3.6.1.2.1.2.2.1.3.4	6
MIB2.4.21.1.3.0.0.0.0	1		.1.3.6.1.2.1.2.2.1.3.5	131
MIB2.4.21.1.3.169.254.0.0	0		.1.3.6.1.2.1.2.2.1.4.1	16436
MIB2.4.21.1.3.192.168.136.0	0		.1.3.6.1.2.1.2.2.1.4.2	1500
MIB2.4.21.1.7.0.0.0.0	192.168.136.1		.1.3.6.1.2.1.2.2.1.4.3	1500
MIB2.4.21.1.7.169.254.0.0	0.0.0.0		.1.3.6.1.2.1.2.2.1.4.4	1500
MIB2.4.21.1.7.192.168.136.0	0.0.0.0		.1.3.6.1.2.1.2.2.1.4.5	1480
MIB2.4.21.1.8.0.0.0.0	4		.1.3.6.1.2.1.2.2.1.5.1	10000000
MIB2.4.21.1.8.169.254.0.0	3		.1.3.6.1.2.1.2.2.1.5.2	100000000
MIB2.4.21.1.8.192.168.136.0	3		.1.3.6.1.2.1.2.2.1.5.3	10000000
MIB2.4.21.1.9.0.0.0.0	2		.1.3.6.1.2.1.2.2.1.5.4	10000000
MIB2.4.21.1.9.169.254.0.0	2		.1.3.6.1.2.1.2.2.1.5.5	0
MIB2.4.21.1.9.192.168.136.0	2		.1.3.6.1.2.1.2.2.1.6.1	
MIB2.4.21.1.11.0.0.0.0	0.0.0.0		.1.3.6.1.2.1.2.2.1.6.2	
MIB2.4.21.1.11.169.254.0.0	255.255.0.0		.1.3.6.1.2.1.2.2.1.6.3	
MIB2.4.21.1.11.192.168.136.0	255.255.255.0		.1.3.6.1.2.1.2.2.1.6.4	
MIB2.4.21.1.13.0.0.0.0	.0.0		.1.3.6.1.2.1.2.2.1.6.5	
MIB2.4.21.1.13.169.254.0.0	.0.0		.1.3.6.1.2.1.2.2.1.7.1	1
MIB2.4.21.1.13.192.168.136.0	.0.0		.1.3.6.1.2.1.2.2.1.7.2	1
MIB2.4.22.1.1.2.192.168.136.1	2		.1.3.6.1.2.1.2.2.1.7.3	2
MIB2.4.22.1.2.2.192.168.136.1			.1.3.6.1.2.1.2.2.1.7.4	2
MIB2.4.22.1.3.2.192.168.136.1	192.168.136.1		.1.3.6.1.2.1.2.2.1.7.5	2
MIB2.4.22.1.4.2.192.168.136.1	3		.1.3.6.1.2.1.2.2.1.8.1	1

On considère l’extrait de la MIB 2 suivant :

```
ipRouteTable OBJECT-TYPE
              SYNTAX  SEQUENCE OF IpRouteEntry
              ACCESS  not-accessible
              STATUS  mandatory
              DESCRIPTION
                      "This entity's IP Routing table."
              ::= { ip 21 }

ipRouteEntry OBJECT-TYPE
              SYNTAX  IpRouteEntry
              ACCESS  not-accessible
              STATUS  mandatory
              DESCRIPTION
                      "A route to a particular destination."
              INDEX   { ipRouteDest }
              ::= { ipRouteTable 1 }
IpRouteEntry ::=
              SEQUENCE {
                  ipRouteDest IpAddress,
                  ipRouteIfIndex INTEGER,
                  ipRouteMetric1 INTEGER,
                  ipRouteMetric2 INTEGER,
                  ipRouteMetric3 INTEGER,
                  ipRouteMetric4 INTEGER,
                  ipRouteNextHop IpAddress,
                  ipRouteType INTEGER,
                  ipRouteProto INTEGER,
                  ipRouteAge INTEGER,
                  ipRouteMask IpAddress,
                  ipRouteMetric5 INTEGER,
                   ipRouteInfo OBJECT IDENTIFIER
                   }

ipRouteDest OBJECT-TYPE
              SYNTAX  IpAddress
              ACCESS  read-write
              STATUS  mandatory
              DESCRIPTION
 "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."
              ::= { ipRouteEntry 1 }

ipRouteIfIndex OBJECT-TYPE
              SYNTAX  INTEGER
              ACCESS  read-write
              STATUS  mandatory
              DESCRIPTION
"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."
              ::= { ipRouteEntry 2 }

ipRouteMetric1 OBJECT-TYPE
              SYNTAX  INTEGER
              ACCESS  read-write
              STATUS  mandatory
              DESCRIPTION
  "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."
              ::= { ipRouteEntry 3 }

ipRouteMetric2 OBJECT-TYPE
              SYNTAX  INTEGER
              ACCESS  read-write
              STATUS  mandatory
              DESCRIPTION
 "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."
              ::= { ipRouteEntry 4 }
[...]
ipRouteNextHop OBJECT-TYPE
              SYNTAX  IpAddress
              ACCESS  read-write
              STATUS  mandatory
              DESCRIPTION
                      "The IP address of the next hop of this route.
                      (In the case of a route bound to an interface
                      which is realized via a broadcast media, the value
                      of this field is the agent's IP address on that
                      interface.)"
              ::= { ipRouteEntry 7 }
ipRouteType OBJECT-TYPE
              SYNTAX  INTEGER {
                          other(1),        -- none of the following
                          invalid(2),      -- an invalidated route
                                           -- route to directly
                          direct(3),       -- connected (sub-)network
                                           -- route to a non-local
                          indirect(4)      -- host/network/sub-network
                      }
              ACCESS  read-write
              STATUS  mandatory
              DESCRIPTION
 "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 dissasociates the
destination identified with said entry from the route identified with said
entry.  It is an implementation-specific matter as to whether the agent removes
an invalidated entry from the table. Accordingly, management stations must be
prepared to receive tabular information from agents that corresponds to entries
not currently in use. Proper interpretation of such entries requires examination
of the relevant ipRouteType object."
              ::= { ipRouteEntry 8 }

ipRouteProto OBJECT-TYPE
              SYNTAX  INTEGER {
                          other(1),       -- none of the following
                                          -- non-protocol information,
                                          -- e.g., manually configured
                          local(2),       -- entries
                                          -- set via a network
                          netmgmt(3),     -- management protocol
                                          -- obtained via ICMP,
                          icmp(4),        -- e.g., Redirect
                                          -- the remaining values are
                                          -- all gateway routing
                                          -- protocols
                          egp(5),
                          ggp(6),
                          hello(7),
                          rip(8),
                          is-is(9),
                          es-is(10),
                          ciscoIgrp(11),
                          ospf(13),
                          bgp(14)
                      }
              ACCESS  read-only
              STATUS  mandatory
              DESCRIPTION
 "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."
              ::= { ipRouteEntry 9 }
          ipRouteAge OBJECT-TYPE
              SYNTAX  INTEGER
              ACCESS  read-write
              STATUS  mandatory
              DESCRIPTION
"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."
              ::= { ipRouteEntry 10 }
ipRouteMask OBJECT-TYPE
              SYNTAX  IpAddress
              ACCESS  read-write
              STATUS  mandatory
              DESCRIPTION
 "Indicate the mask to be logical-ANDed 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.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."
              ::= { ipRouteEntry 11 }
[...]
ipRouteInfo OBJECT-TYPE
              SYNTAX  OBJECT IDENTIFIER
              ACCESS  read-only
              STATUS  mandatory
              DESCRIPTION
 "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 recognize this value."
              ::= { ipRouteEntry 13 }
```