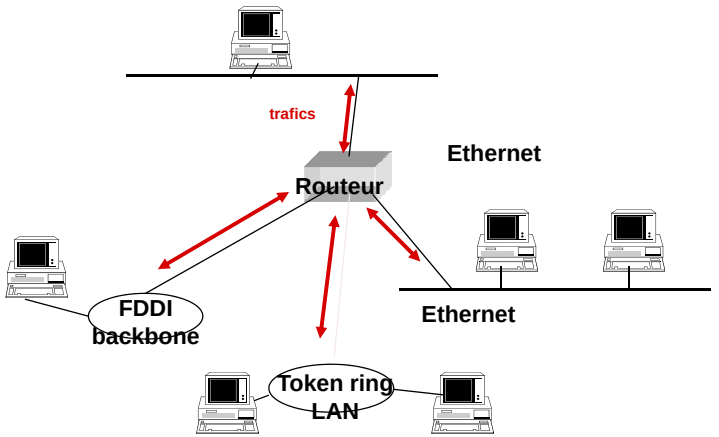
Administration et sécurité des réseaux

Chapitre 2

Le Protocole SNMP
(Simple Network Management Protocol)

SNMP : Motivation

- Nécessité d 'avoir un protocole permettant de remonter des informations sur l 'activité des différentes ressources du réseau (les serveurs, les routeurs, les hubs, etc).



Présentation de SNMP

- Protocole d'administration de machines supportant TCP/IP
  - SNMP Version 1 (SNMPv1) Défini dans la RFC 1157
    - Mécanisme de sécurité basé sur la notion de communauté (mot de passe en clair dans les requêtes et réponses)
  - SNMP Version 2 (SNMPv2) Défini dans les RFC 1905, 1906 et 1907
    - Introduit deux nouveaux types de paquets get-bulk-request et inform-request (communication entre plate-formes)
  - SNMP Version 3 (SNMPv3) Défini dans les RFC 2570, 2571, 2572, 2573, 2574 et 2575
    - Introduit de nouveaux mécanismes de sécurité (authentification forte et confidentialité)

Présentation de SNMP

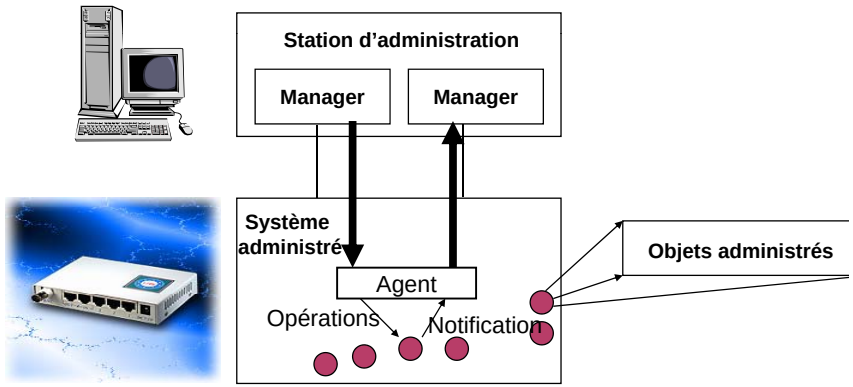
- Répond à un grand nombre de besoins :
  - Administrer à distance des machines indépendamment de leur architecture
  - Disposer d'une cartographie du réseau
  - Fournir un inventaire précis de chaque machine
  - Mesurer la consommation d'une application
  - Signaler les disfonctionnements

Modèle d 'administration SNMP

- Une administration SNMP est composée de trois types d'éléments :
  - des **agents** chargés de superviser un équipement. On parle d'agent SNMP installé sur tout type d'équipement.
  - une ou plusieurs **stations de gestion** capables d'interpréter les données
  - une **MIB** (Management Information Base) décrivant les informations gérées (objets administrés).
- SNMP permet **la supervision, le contrôle et la modification** des paramètres des éléments du réseau.

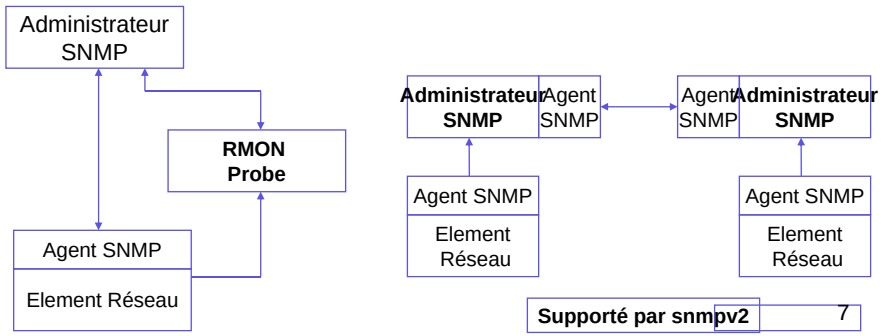
Modèle d 'administration des réseaux

- Le modèle « Manager-Agent » ou modèle deux-tiers.



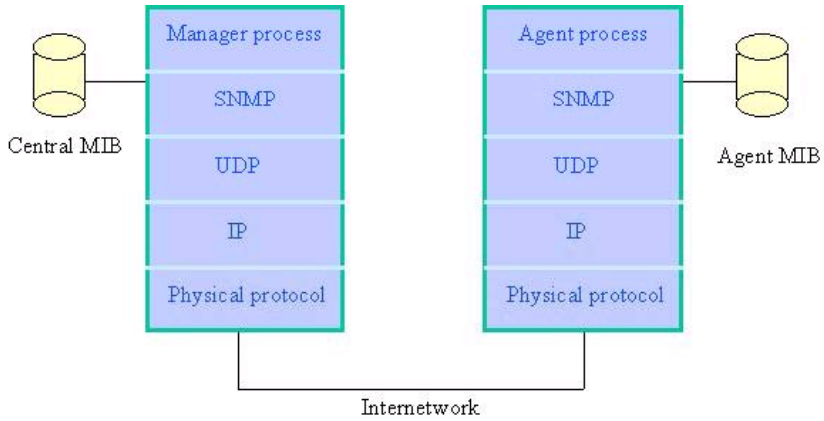
Modèle d 'administration SNMP

- L 'architecture **trois-tiers** insère entre le Manager et l 'agent une sonde RMON ou une autre station d 'administration (modèle SNMPv2).
  - La sonde RMON permet de faire la collecte d 'informations d 'administration et quelques traitements sur le trafic.



L 'architecture de SNMP

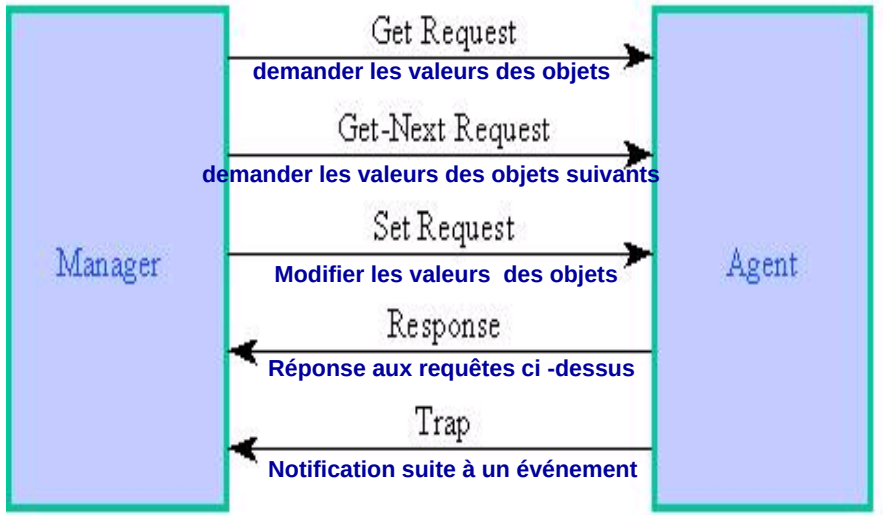
- SNMP fonctionne au dessus de UDP



Les opérations SNMP

- SNMP offre 3 opérations simples :
  - **GET :**
    - ➔ Permet à la station d'administration de retirer les valeurs d'un objet de la station administrée.
  - **SET:**
    - ➔ Permet à la station d'administration d'affecter des valeurs à un objet dans la station administrée.
  - **TRAP:**
    - ➔ Permet à une station administrée d'envoyer des notifications à la station d'administration pour les événements significatifs.

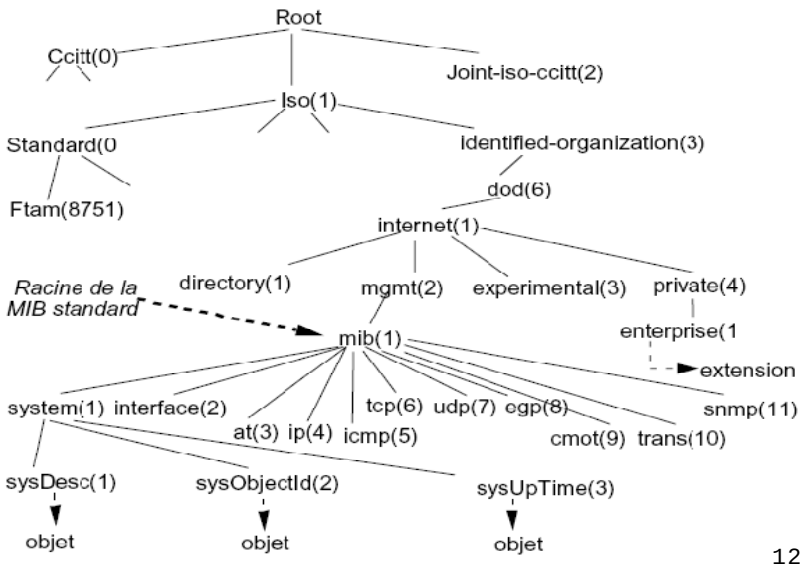
Les PDUs SNMP



La MIB (Management Information Base)

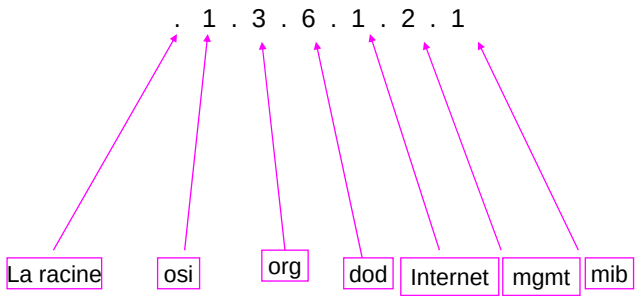
- 1 ressource à gérer = 1 objet
- Les objets administrables sont une abstraction des ressources physiques (interfaces, équipements,etc.) et logiques (connexion TCP, paquets IP, etc.)
- **MIB** : collection structurée d'objets reconnus par les agents
- Chaque nœud dans le système doit maintenir une MIB qui reflète l'état des ressources gérées
  - Une entité d'administration peut accéder aux ressources du nœud en lisant les valeurs de l'objet ou en les modifiant
- **MIB: 2 objectifs**
  - Un schéma commun : SMI (Structure of Management Information)
  - Une définition commune des objets et de leur structure

Arbre des MIB accessibles

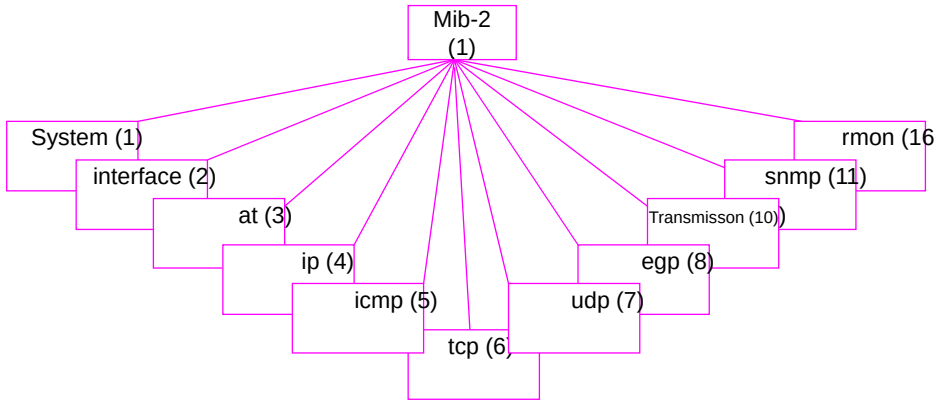


Identificateur d 'un objet de la MIB

- Identificateur d 'un objet:
  - ➔ Identificateur unique = séquence d 'entiers dont chacun représente la position de ces successeurs dans l 'arbre.
  - Exemple**: identificateur de l 'objet MIB** :



Le groupe MIB-2



Le groupe MIB-2

MIB-2

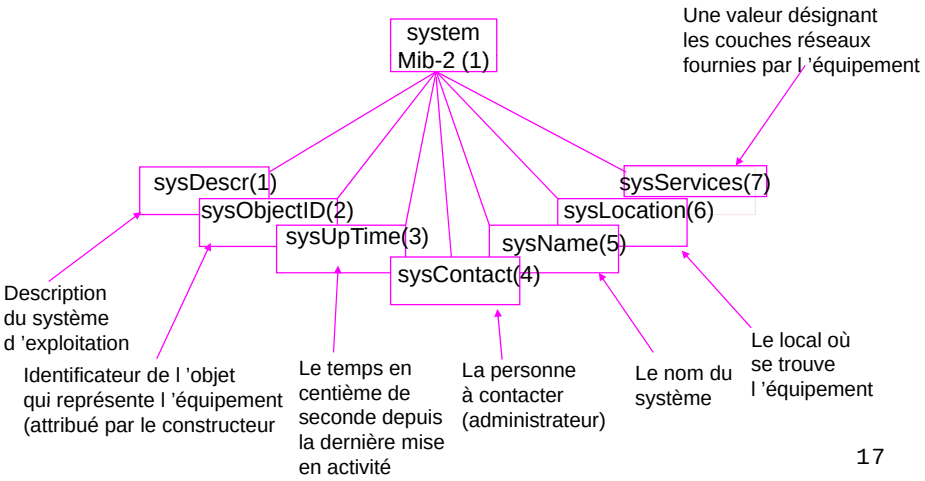
groupe	nbre éléments	commentaire
system	7	nœud dans le réseau
interfaces	25	interfaces réseau
at	5	IP address translation
ip	65	Internet Protocol
icmp	26	Internet Control Message Protocol
tcp	21	Transmission Control Protocol
udp	8	User Datagram Protocol
egp	22	Exterior Gateway Protocol
transmission	114	informations sur la transmission
snmp	28	SNMP
rmon	218	Remote network monitoring

La structure numérique de la MIB-2

system	1.3.6.1.2.1.1
interfaces	1.3.6.1.2.1.2
at	1.3.6.1.2.1.3
ip	1.3.6.1.2.1.4
icmp	1.3.6.1.2.1.5
tcp	1.3.6.1.2.1.6
udp	1.3.6.1.2.1.7
egp	1.3.6.1.2.1.8
rmon	1.3.6.1.2.1.9
transmission	1.3.6.1.2.1.10
snmp	1.3.6.1.2.1.11

Le groupe « System »

- **System** : correspond au nom de l'agent, n° de version, type de la machine, nom du système d'exploitation, etc.



Le groupe « Interface »

ifNumber : le nombre d 'interfaces

ifIndex : Index de l 'interface (son numéro)

ifDescre : Description de l 'interface

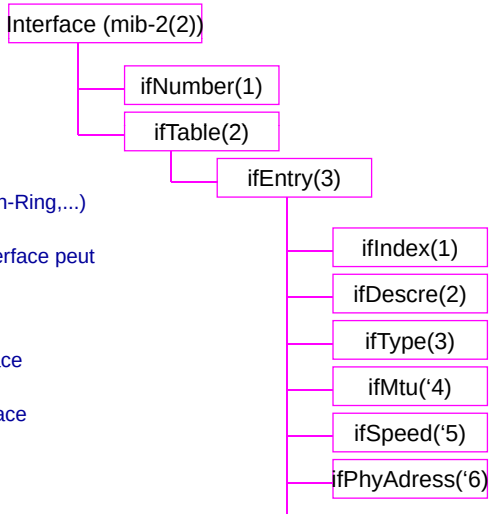
ifType : le type de l 'interface (Ethernet, Token-Ring,...)

ifMtu : le nombre maximum d 'octet que l 'interface peut envoyer ou recevoir

ifSpped : Une estimation du débit de l 'interface

ifPhyAdress : l 'adresse physique de l 'interface

....



Le groupe « IP »

ipForwarding : Agit comme passerelle, ou non

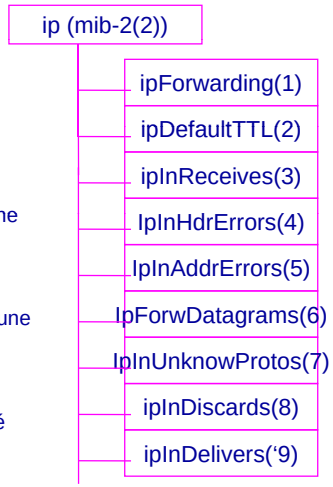
ipDefault TTL : la valeur par défaut du TTL ajouté dans un paquet IP

ipInReceives : Le nombre total de paquets IP reçus

IpInHdrErrors : Le nombre total de paquets écartés dus à une erreur sur l 'en-tête

IpInAddrErrors : Le nombre total de paquets écartés dus à une erreur sur l 'adresse de destination

IpForwDatagrams : Le nombre total de paquets dont l 'entité réceptrice ne représente pas la destination finale.



Les autres groupes

icmp : 26 compteurs

- pour chaque message icmp, 2 compteurs pour compter les messages reçus et émis
- 4 compteurs pour compter le nombre total de messages icmp reçus, reçus par erreur ou non envoyés,

tcp : rend compte des connexions TCP en cours et leurs paramètres

de type nombre max de connexions simultanées permises, nombre d'ouvertures actives, l'état de chaque connexion (écoute, time-wait,...).

udp : - 4 compteurs renseignent sur le nombre de datagramme

UDP envoyés, reçus, en erreur, ...

egp : gère le protocole egp (External gateway protocol)(routage

des paquets entre routeurs). On a le nbre de paquets entrants, sortants, en erreur, la table des routeurs adjacents, des infos sur les routeurs...

snmp : requis pour chaque entité mettant en oeuvre le protocole

SNMP. Contient le nombre de messages SNMP entrants et sortants, le nombre de mauvaises versions reçues ou de nom de communauté invalide, la répartition du type de requêtes reçues et envoyées (get, get_next, set et trap)

Structure des informations d 'Administration (SMI)

- La MIB contient des éléments simples (scalaire et tableaux à deux dimensions de scalaires)
- **SMI** (Structure of Management information) : donne les règles de **définition, d'accès et d'ajout** des objets dans la MIB (méta-modèle)
- **Objectifs** : encourager la simplicité et l'extension de la base d 'informations d'Administration :
  - Représentation identique des objets → rendre un objet accessible de la même manière sur chaque entité du réseau
- L 'objet administré peut être considéré d 'être composé d 'un type d 'objet et une instance.
- SMI définit le type d 'objets et non leur instance.

Structure des informations d 'Administration (SMI)

- Un objet possède :
  - un nom (Descripteur + identificateur d 'objet)
  - une syntaxe utilisant ASN.1 (Abstract Syntaxe Notation)
  - une définition qui est un texte de description de l 'objet
  - un accès qui spécifie les droits d 'accès à l 'objet (read only, read-write or not accessibe)
  - Un statut qui spécifie si l 'objet est courant (mondatory ou optional) ou obsolète.
  - un schéma de codage BER (Basic Encoding Rules)

Structure des informations d 'Administration (SMI)

- Les caractéristiques d 'un objet sont regroupées dans la définition d 'une macro qui définie la structure d 'un type d 'objet :

```
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"
      |"deprecated"
   END
```


Structure des informations d 'Administration (SMI)

```
atIndex OBJECT-TYPE
              SYNTAX  Integer
              ACCESS  read-write
              STATUS  mandatory
              DESCRIPTION « Numéro d'interface logique."
              ::= { atEntry 1 }
atIndex OBJECT-TYPE
              SYNTAX  Integer
              ACCESS  read-write
              STATUS  mandatory
              DESCRIPTION « Numéro d'interface logique."
              ::= { atEntry 1 }
```

Structure des informations d 'Administration (SMI)

- La structure de types de données ASN.1 pour SNMP :

Structure	type de données	commentaires
types primitives	INTEGER	donnée de type entier
	OCTET STRING	données de type caractère ayant une taille de 8 bits
	OBJECT IDENTIFIER	Position d'un objet dans la MIB
	NULL	objet sans type
Types dérivés ou applicatifs	NetworkAdress	Non utilisé
	IpAdress	une adresse IP
	Counter	un entier non négatif qui croît d'une façon monotone et ayant un maximum égal à 2^{32}-1
	Gauge	un entier non négatif qui peut croître ou décroître
	TimeTicks	un entier non négatif en centième de seconde comme unité
types constructeur	SEQUENCE	Une liste d'objets
	SEQUENCE OF	un tableau d'objets

Mécanismes de sécurité de SNMP

- SNMP implémente 3 mécanismes de sécurité:
  - L 'authentification,
  - l 'autorisation (politique d 'accès)
  - L 'identification de l 'objet
- L 'authentification se fait par le choix d 'un nom de communauté afin de restreindre l 'accès aux agents que par les administrateurs réseaux.
  - Le nom de communauté est vérifié pour chaque requête SNMP.
  - Il est relié au mode d 'accès aux objets de la MIB (lecture-écriture).
- Chaque communauté définie un mode d 'accès qui peut être soit Read-only, soit read-write.

Mécanismes de sécurité de SNMP

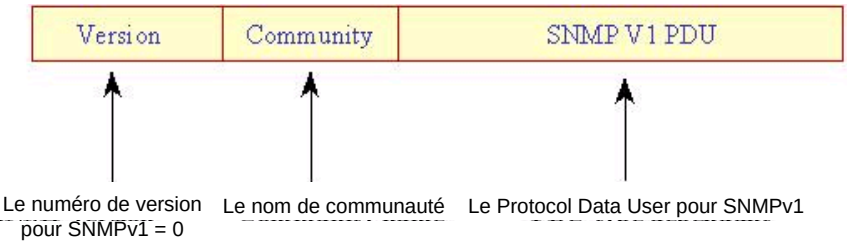
- L 'autorisation est l 'intersection entre le mode d 'accès défini par la communauté et l 'accès à l 'objet défini parmi les caractéristiques de l 'objet.

Mode d'accès	read-only	read-write	write-only	not-accessible
read-only	3	3	1	1
read-write	3	2	4	1

- où les classes sont définies par :

1 no right
2 get, get-next, set, trap
3 get, get-next, trap
4 set, trap

Format général du Message SNMP



- SNMP community = Un ensemble d'administrateurs autorisés à utiliser l'agent
- Chaque communauté est définie en utilisant un nom unique
- Les administrateurs doivent préciser le nom de la communauté dans les requêtes SNMP

Définition ASN.1 du Message

RFC1157-SNMP DEFINITIONS ::= BEGIN

IMPORTS *ObjectName*, *ObjectSyntax*, ... FROM *RFC1155-SMI*;

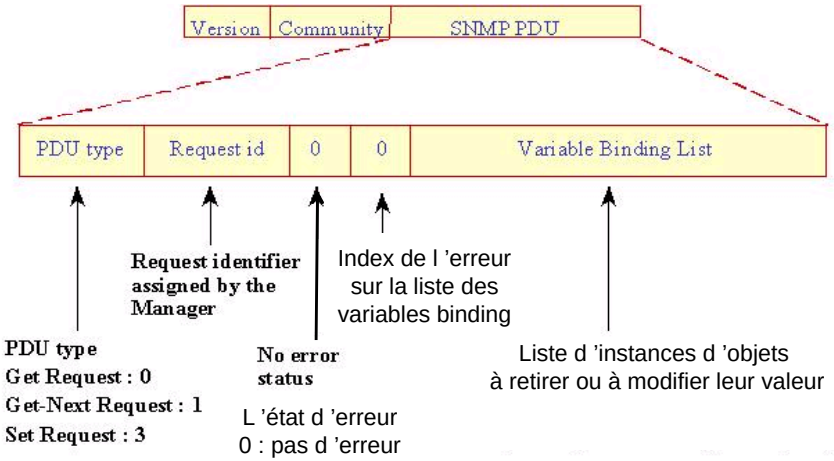
Message ::= SEQUENCE {

version INTEGER, — Version

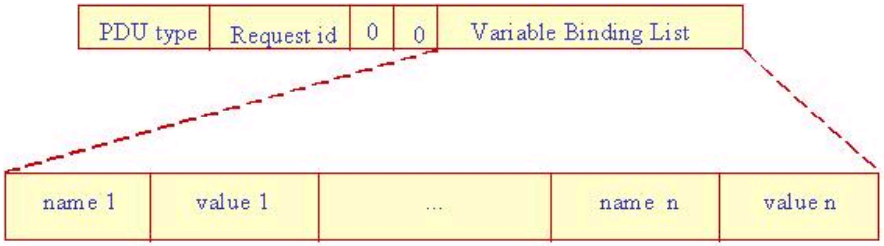
community OCTET STRING, — Community

data ANY) — SNMP PDU

Format des Get, Get-Next et Set



Format de Variable Binding List



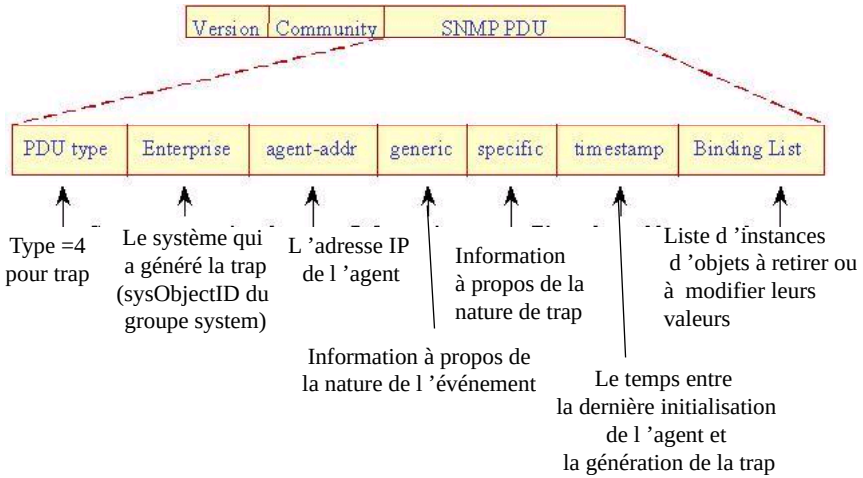
VarBind ::= **SEQUENCE** {

name ObjectName,

value ObjectSyntax }

VarBindList ::= **SEQUENCE OF** *VarBind*

Format de Trap



Le champ “Generic”

- Le champ “Generic” peut prendre une des valeurs suivantes :
  - **coldStart (0) :** Une réinitialisation inatendue due à une défaillance.
  - **warmStart (1)** : Une défaillance mineur
  - **linkDown (2)** : Une défaillance survenue sur une interface physique.
  - **linkUp (3)** : Une interface devient active.
  - **authenticationFailure (4) :** L’agent a reçu un message avec une authentification impropre
  - **egpNeighborLoss (5) :** Un routeur voisin utilisant EGP (External Gateway Protocol) est déclaré comme étant non focntionnel
  - **enterpriseSpecific (6) :** L’événement relatif à “enterprise-specific” est survenu

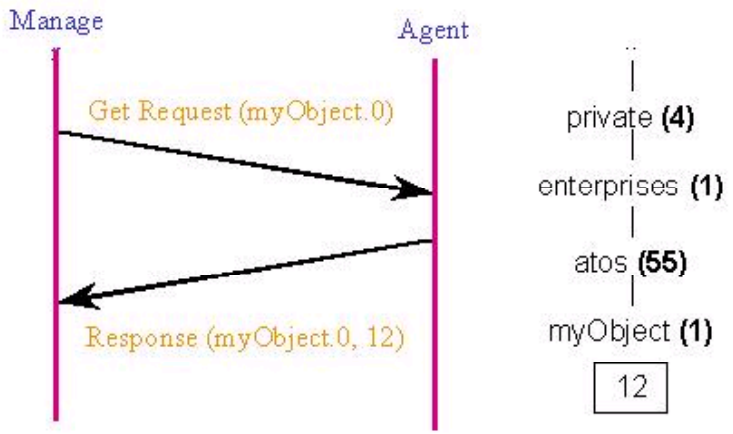
Exemple de Trap

Trap	Enterprise	agent-addr	generic	specific	timestamp
4	1.3.6.1.4.1.20.1	132.18.54.21	3	0	22759400
ipInReceives.0		956340			

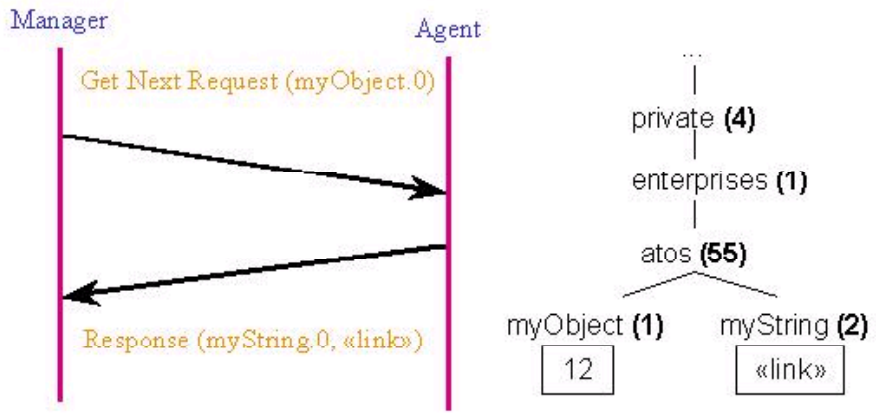
Binding List

- L ’adresse IP de agent émetteur : 132.18.54.21
- L ’objet concerné par la trap est : 1.3.6.1.4.1.20.1 (MIB privée)
- Type de trap : link up (generic=3)
- Indication : le nombre de paquets reçus est 956340
- La dernière réinitialisation de l’agent : 6 heures passées.

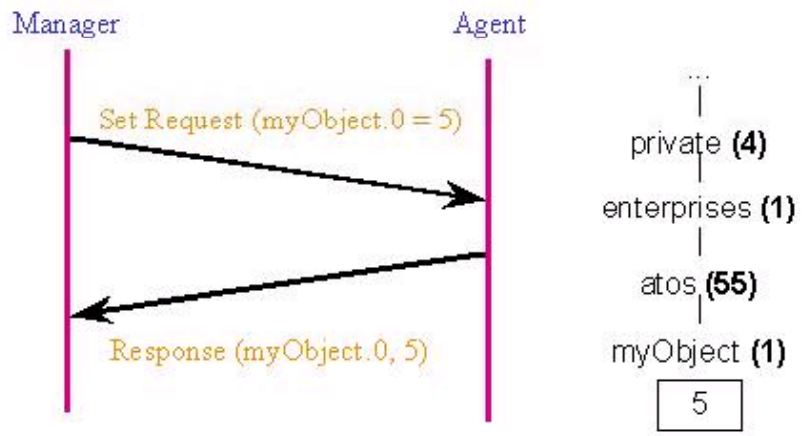
La requête GET



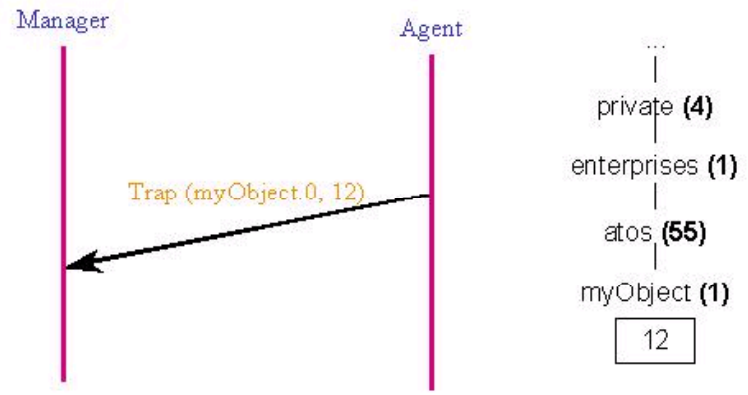
La requête GETNextRequest



La requête Set

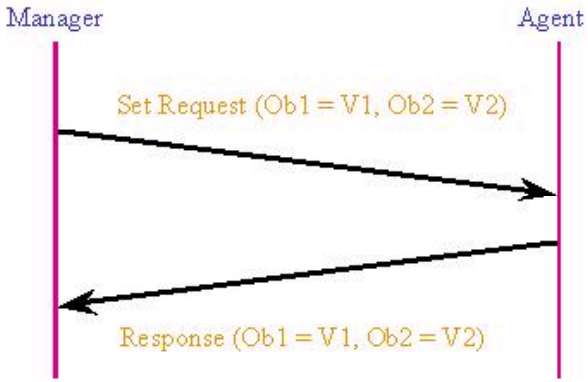


La notification TRAP

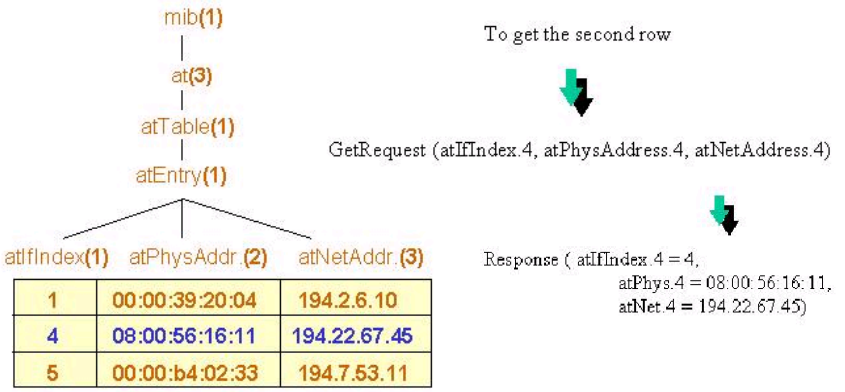


Les requêtes multiples

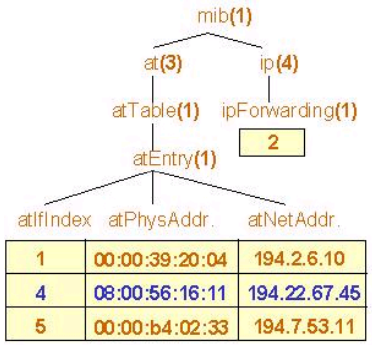
- Les requêtes Get, Get Next and Set Requests peuvent préciser plusieurs objets à lire ou à modifier leurs valeurs.



Exemple de Get Request



Exemple de GetNext Request

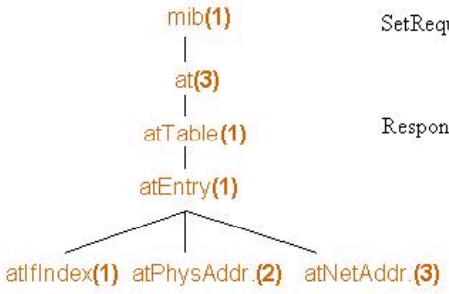


1	00:00:39:20:04	194.2.6.10
4	08:00:56:16:11	194.22.67.45
5	00:00:b4:02:33	194.7.53.11

GetNextRequest (atIfIndex.1, atPhys.1, atNet.1)

Response (atIfIndex.4 = 4,
atPhys.4 = 08:00:56:16:11,
atNet.4 = 194.22.67.45)

Exemple de Set Request

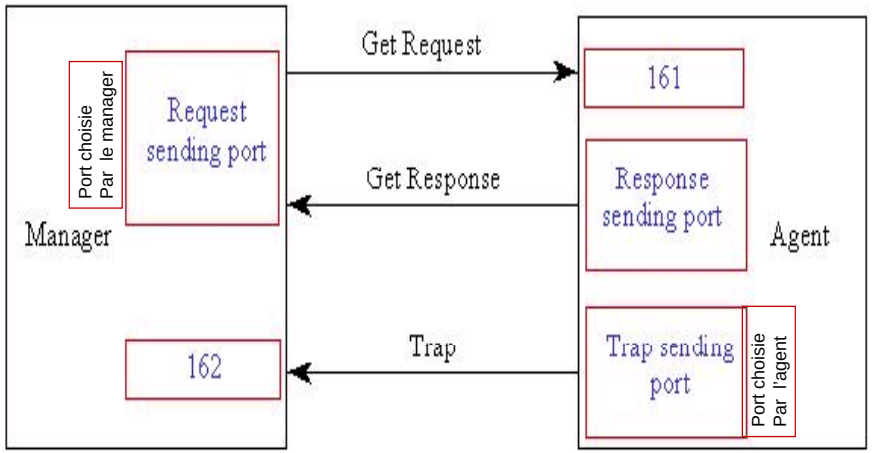


1	00:00:39:20:04	194.2.6.10
4	00:00:77:b1:45	194.22.67.45
5	00:00:b4:02:33	194.7.53.11

SetRequest (atPhysAddress.4 = 00:00:77:b1:45)

Response (atPhysAddress.4 = 00:00:77:b1:45)

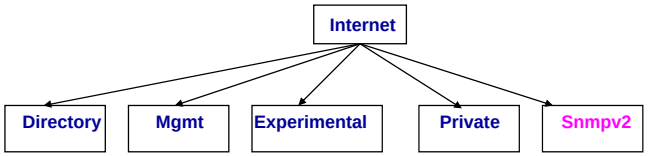
Numéros des Ports de SNMP



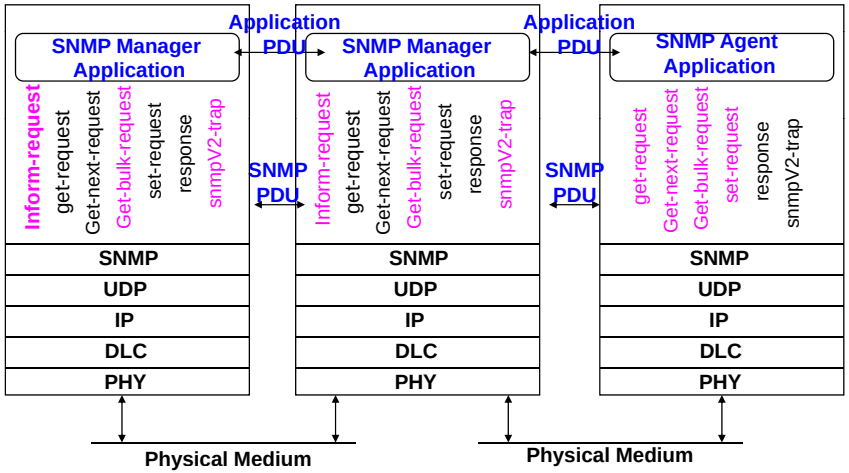
SNMPv2

Introduction

- SNMPv2: mêmes éléments de base que SNMPv1
- Différence significative:
  - ➔ un agent et un manager ont la même fonction.
  - ➔ Deux messages sont ajoutés :
    - **get-bulk**: demander et recevoir un grand volume de données
    - **Inform request**: pour la communication entre deux systèmes d 'administration



L 'architecture de SNMPv2



Les opérations de SNMPv2

- Les messages **get-request, get-next-request, et set-request** sont les mêmes que ceux de SNMPv1 et ils sont générés par l 'application d 'administration.
- Le message **response** est le même aussi que celui de SNMPv1, mais il est généré dans ce cas par l 'agent ou le manager.
- Le message **inform-request** est généré par le manager et envoyé à un autre manager.
- Le message **get-bulk-request** est généré par le manager afin de transférer une grande quantité de données de l 'agent vers le manager.
- L 'événement **SNMPv2-trap** (notification) est généré et transmis quand une situation exceptionnelle apparaît.

Les opérations de SNMPv2

- La structure de données PDU dans SNMPv2 a été uniformisée pour tous les messages (sauf pour le message get-bulk-request) afin d 'améliorer les performances d 'échange.

PDU Type	RequestID	Error Status	Error Index	VarBind 1 name	VarBind 1 value	...	VarBind n name	VarBind n value

  - avec SNMPv1 Les VarBinds ne sont pas toutes retournées dans le cas d 'une erreur (Error Status ≠ 0), avec SNMPv2 uniquement la varBind qui génère l 'erreur est ignorée et le reste sera retournée dans la réponse.
- La structure de données PDU get-bulk-request est :

PDU Type	RequestID	Non-Repeaters	Max Repetitions	VarBind 1 name	VarBind 1 value	...	VarBind n name	VarBind n value

  - non-repeaters : nombre de variables non répétés à retourner (variables atomiques)
  - max-repeaters : nombre de lignes à retourner (variables composées)
- get-next-request ne peut retourner qu'une seule ligne (la ligne qui suit celle précisée par les varBinds)

SMI2 de SNMPv2

- **Un ensemble supplémentaire** de la SMI de SNMP V1 : même hiérarchie de nommage, même identification d'objet.
- SMIv2 (RFC 1902) introduit trois concepts clés :
  - Objet et définition de Tables (utilisant la macro OBJECT-TYPE V2)
  - définition de Traps (utilisant la macro NOTIFICATION-TYPE V2)
  - définition de Modules (utilisant la macro MODULE-IDENTITY V2)
- Les différences entre SNMP V1 et SNMP V2 concernent :
  - Les types d'objet (universal et application-wide types)
  - la macro OBJECT-TYPE utilisée pour définir les objets

SMI2 de SNMPv2

- Les types universels de SMI-1 et SMI-2

Object Types	SNMP V1	SNMP V2
INTEGER	X	X
OCTET STRING	X	X
OBJECT IDENTIFIER	X	X
NULL	X	

Constructor Types	SNMP V1	SNMP V2
SEQUENCE, SEQUENCE OF	X	X

SMI2 de SNMPv2

Object Types	SNMP V1	SNMP V2
IP Address	X	X
Counter32	X	X
Counter64		X
Unsigned32		X
Gauge32	X	X
TimeTicks	X	X
Opaque	X	X
BITS		X

SMI2 de SNMPv2

- Définitions Associées à la Macro OBJECT-TYPE

ObjectName ::= OBJECT IDENTIFIER

ObjectSyntax ::= CHOICE { INTEGER, OCTET STRING,
OBJECT IDENTIFIER,
IpAddress, Counter32,
TimeTicks, Opaque, Gauge32,
Counter64, Unsigned32 }

SNMP V2

ObjectName ::= OBJECT IDENTIFIER

ObjectSyntax ::= CHOICE { INTEGER, OCTET STRING,
OBJECT IDENTIFIER, NULL,
IpAddress, Counter,
TimeTicks, Opaque, Gauge }

SNMP V1

SMI2 de SNMPv2

```
OBJECT-TYPE MACRO ::= BEGIN
   TYPE NOTATION ::=    «SYNTAX» type (ObjectSyntax)
                        UnitsPart
                        «MAX-ACCESS» Access
                        «STATUS» Status
                        DescrPart  ReferPart
                        IndexPart  DefValPart
   VALUE NOTATION ::= value (ObjectName)
```

SNMP V2

```
OBJECT-TYPE MACRO ::= BEGIN
   TYPE NOTATION ::=    «SYNTAX» type (ObjectSyntax)
                        «ACCESS» Access
                        «STATUS» Status
                        DescrPart  ReferPart
                        IndexPart  DefValPart
   VALUE NOTATION ::= value (ObjectName)
```

SNMP V1

SMI2 de SNMPv2

- La clause **units** est une clause optionnelle qui indique l'unité associée avec l'objet (time, length, ...)

```
UnitsPart ::= «UNITS» Text | empty
Text ::= « value (OCTET STRING) »
```

SNMP V2

- La clause DEFVAL est la même pour les deux verisons

```
DefValPart ::=  «DEFVAL» «{» value (ObjectSyntax) «}»
                | empty
```


SMI2 de SNMPv2

- La clause MAX-ACCESS :

```
«MAX-ACCESS» Access
Access ::= «read-only» | «read-write» | «read-create»
           | «not-accessible» | «accessible-for-notify»
```

SNMP V2

read-only	: read access
read-write	: read and write acceses
read-create	: read, write and create accesses
not-accessible	: not-accessible for any operation
accessible-for-notify	: accessible only via a notification (trap)

```
«ACCESS» Access
Access ::= «read-only» | «read-write» | «write-only» |
           «not-accessible»
```

SNMP V1

SMI2 de SNMPv2

- La clause STATUS :

```
«STATUS» Status
Status ::= «current» | «obsolete» | «deprecated»
```

SNMP V2

current	: the object is valid for the current standard
obsolete	: the object should not be implemented
deprecated	: the object is obsolete, but may be implemented to provide interoperability with older implementations

```
«STATUS» Status
Status ::= «mandatory» | «optional» | «obsolete»
           | «deprecated»
```

SNMP V1

SMI2 de SNMPv2

- La clause DESCRIPTION : même clause comme SNMPv1, mais obligatoire pour SNMPv2

```
DescrPart ::= «DESCRIPTION» Text
Text ::= « OCTET STRING »
```

SNMP V2

```
DescrPart ::= «DESCRIPTION» value (DisplayString)
                | empty
```

SNMP V1

SMI2 de SNMPv2

- (MIB-2) : Exemple d'Instance OBJECT-TYPE

```
ifTestResult OBJECT-TYPE
    SYNTAX INTEGER
      { none(1), success(2), inProgress(3), notSupported(4),
        unAbleToRun(5), aborted(6), failed(7) }
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION «Result of the most recently requested test»
    ::= { ifTestEntry 4 }
```

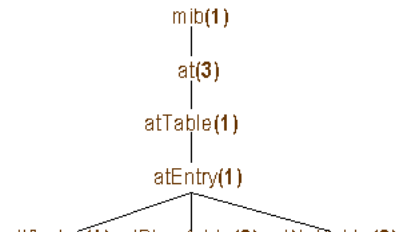

SMI2 de SNMPv2

- Exemple de table :

```
atTable OBJECT-TYPE
    SYNTAX SEQUENCE OF AtEntry
    MAX-ACCESS not-accessible
    STATUS deprecated
    DESCRIPTION «Translation table»
    ::= { at 1 }

atEntry OBJECT-TYPE
    SYNTAX AtEntry
    MAX-ACCESS not-accessible
    STATUS deprecated
    DESCRIPTION «Translation entry»
    INDEX {atIfIndex, atPhysAddress}
    ::= { atTable 1 }

AtEntry ::= SEQUENCE {
    atIfIndex       INTEGER,
    atPhysAddress   OCTET STRING,
    atNetAddress    IpAddress
    }
```



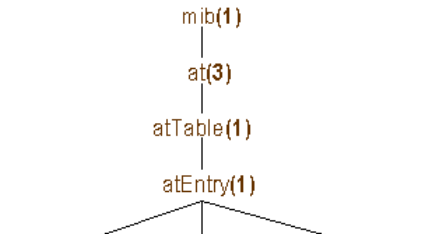
1	00:00:39:20:04	194.2.6.10
4	08:00:56:16:11	194.22.67.45
5	00:00:b4:02:33	194.7.53.11

SMI2 de SNMPv2

```
atIfIndex OBJECT-TYPE
    SYNTAX INTEGER
    MAX-ACCESS read-only
    STATUS deprecated
    DESCRIPTION «Index»
    ::= { atEntry 1 }

atPhysAddress OBJECT-TYPE
    SYNTAX OCTET STRING
    MAX-ACCESS read-only
    STATUS deprecated
    DESCRIPTION «Physical Address»
    ::= { atEntry 2 }

atNetAddress OBJECT-TYPE
    SYNTAX IpAddress
    MAX-ACCESS read-only
    STATUS deprecated
    DESCRIPTION «Network Address»
    ::= { atEntry 3 }
```



1	00:00:39:20:04	194.2.6.10
4	08:00:56:16:11	194.22.67.45
5	00:00:b4:02:33	194.7.53.11

SMI2 de SNMPv2

- SMI-v2 étend le concept des tables afin d 'avoir l 'agrégation d 'une table simple pour obtenir une table multiple.
  - Possibilité d 'augmenter une table par une deuxième table ayant le même nombre d 'éléments (utilisation de la clause augment)

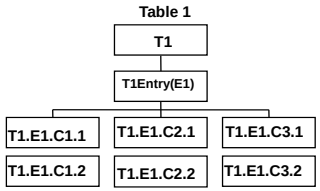


Table de base

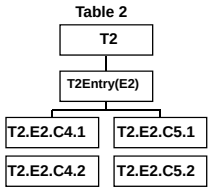


table augmentée

```
T1Entry      OBJECT-TYPE
  SYNTAX              TableT1Entry
  MAX-ACCESS          not-accessible
  STATUS              current
  DESCRIPTION         "An entry in table T1"
  INDEX               {T1.E.C1}
      ::={T1 1}
```

```
T1Entry      OBJECT-TYPE
SYNTAX                TableT2Entry
MAX-ACCESS            not-accessible
STATUS                current
DESCRIPTION           "An entry in table T2"
AUGMENTS{T1Entry}
      ::={T2 1}
```


SMI2 de SNMPv2

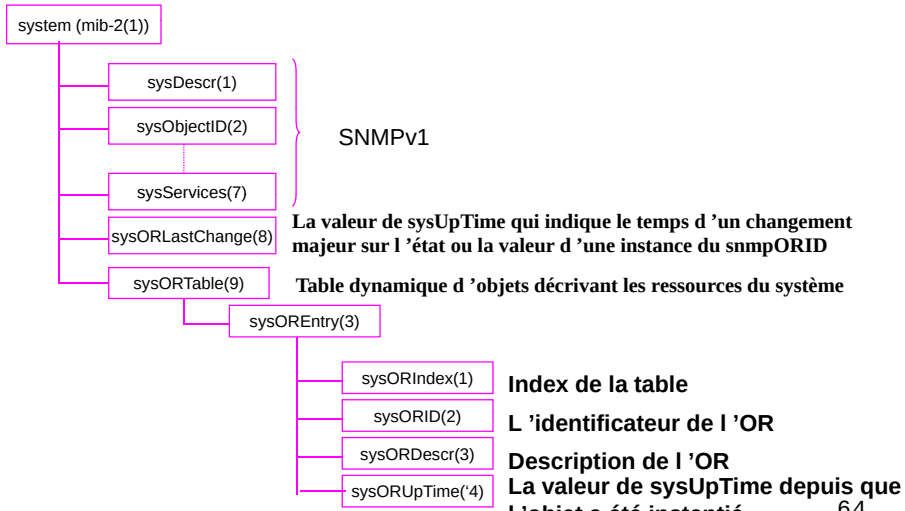
- L 'introduction d 'une nouvelle colonne d 'état appelée RowStatus.
- La création d 'une nouvelle ligne se fait par deux méthodes :
  - Créer une ligne et la mettre active immédiatement : RowStatus=CreateAndGo(4)
  - Créer une ligne et la mettre active ultérieurement : RowStatus=CreateAndWait(5)
- Les opération de création et suppression se fait par la requête Set-request
  - création : SetRequest(RowStatus=4 ou 5)
  - supression : SetRequest (RowStatus=6)

La MIB

- La MIB de SNMPv2 est définie dans la RFC 1907
- Trois nouveaux groupes dansSNMP V2 MIB :
  - *system* group :
    - extension du groupe original "MIB-II *system"*
    - le groupe SNMP V1 system + de nouveaux objets
  - *snmp* group :
    - rafinement du groupe original "MIB-II *snmp"*
    - le groupe SNMP V1 snmp + de nouveaux objets
  - *snmpMIBObjects* group : traite les "SNMPv2-Trap PDUs"
    - snmpTrap subgroup : Informations à propos des traps générés par les agents
    - snmpSet subgroup : Utilisé pour résoudre des problèmes qui proviennent des opérations SET.

La MIB

- Le groupe "system"



La MIB

- Le groupe "snmp"

snmp (mib-2(11))

snmpInPks(1)	**Le nombre total de messages délivrés du service de transport**
SnmpInBadVersions(3)	**Le nombre total de messages du service de transport dont la version est invalide**
snmpInBadCommunityNames(4)	**Le nombre total de messages du service de transport dont le nom de communauté est inconnu**
snmpInBadCommunityUses(5)	**Le nombre total de messages du service de transport dont le nom de communauté est non autorisé par l 'opération**
snmpInASNParseError(6)	**Le nombre total d 'erreurs sur ASN.1 et BER**
snmpEnableAuthenTraps(30)	**Indique si l 'entité SNMPv2 est autorisée à générer des traps pour un échec sur l ' 'authentification.**
snmpsilentDrops(31)	**Le nombre total de requêtes échouées à cause d 'une réponse qui dépasse la taille maximale ou une exception sur varbinds**
snmpProxyDrops(32)	**Le nombre total de requêtes échouées à cause d 'une inaccessibilité avec un proxy**

La MIB

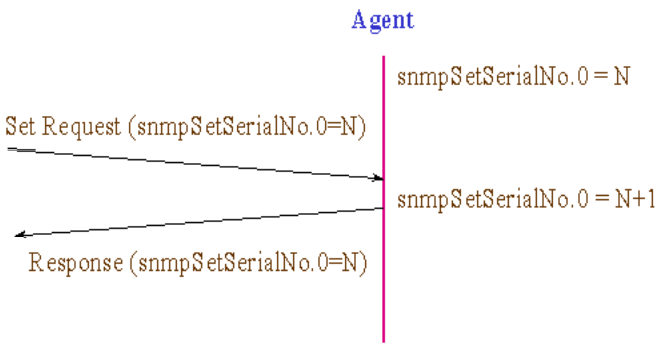
- Le groupe "snmpMIBObjetcs"

SnmpMIBObjects (snmpMIB (1))

- snmpTraps(4)
  - snmpTrapOID (1) — **Identificateur de l 'objet du trap couramment envoyé**
  - snmpTrapEntreprise (3) — **Identificateur de l 'objet de l 'entreprise associé au trap couramment envoyé**
- sysTraps(5)
  - ColdStart(1)
  - warmStart(2)
  - linkDown(3)
  - linkUp(4)
  - autentificationFailure (5)
- snmpSet(6)
  - snmpSetSerialNo(1) — **Verrouillage utilisé pour protéger les objets de la MIB contre des accès concurrents de plusieurs Managers avec l 'opération Set**

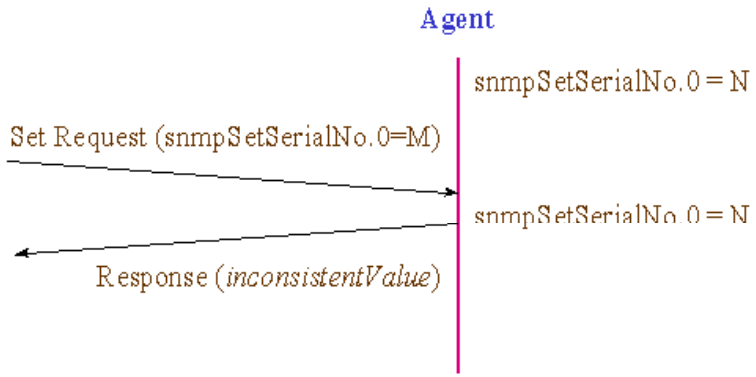
La MIB

- L'agent accepte l'opération SET sur le snmpSetSerialNo si la valeur invoquée est la même que celle de la valeur courante
- la valeur de snmpSetSerialNo est incrémentée de 1

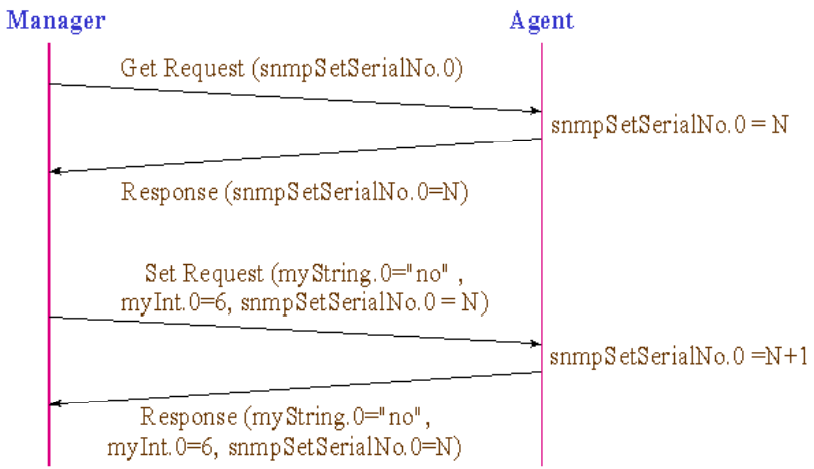


La MIB

- L'agent refuse l'opération SET sur snmpSetSerialNo si la valeur invoquée est différente de la valeur courante



La MIB



Chapitre 3(suite)

Mise en place d'un Agent et d'un Manager SNMP Sous Linux

Mise en place d'un agent SNMP sous Linux

- Implémentation de l'agent SNMP
- **snmpd**
  - démon relatif au service SNMP de base
  - Écoute du port 161
  - Écoute des SET, GET … et génération des Response
- **snmptrapd**
  - Démon responsable de la génération des Trap
  - Gère le port 162

- Agent SNMP activé par activation de ces deux démons
- Configuration possible à travers : **/etc/snmp/snmpd.conf**
- Journalisation et historique : **/var/log/snmpd.log**

Mise en place d'un Manager

- Agent et Manager doivent être dans la même communauté.
- Ensemble d'outils permettant l'exécution de requêtes SNMP

- Exemple : les outils NET-SNMP
  - Intérrogation d'un agent :
    - **snmpget**
    - **snmpgetnext**
    - **snmpwalk**
  - Modification d'attributs de la MIB d'un agent : **snmpset**
  - Outils additionnels : **snmpstat**, **snmptranslate**, **snmpstatus** etc.

  - Manuel d'utilisation, page projet : http://www.net-snmp.org
  - Téléchargement : http://rpm.pbone.net