République Algérienne Démocratique et Populaire

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE SAAD DAHLAB DE BLIDA
FACULTE DES SCIENCES
DEPARTEMENT D'INFORMATIQUE

MEMOIRE DE FIN D'ETUDE

Pour l'obtention du diplôme d'ingénieur d'état en Informatique

Option : **SYSTEME D'INFORMATION**

THEME

Gestion des réseaux

DIPLOMANTS :
M^lle ZENTOR Hanane Kheira
M^r HAMLATI Anis

Structure d'accueil :
SONATRACH

Promoteur:
M^r M.OULD BRAHAM
Encadreur:
M^lle Z.DJAOUDI

Promotion 2002/2003

DEDICACE

A la présentation de ce mémoire, je remercie d'abord le bon dieu le tout puissant de m'avoir donner le courage et la patience pour achever ce modeste travail lequel je dédie à :

A ma Mère, ma première école dans la vie, qui à toujours veille sur mon bien être.

A mon Père, qui m'a aidé pendant toute mon cursus de formation.

A ma sœur Amina pour son soutien moral et matériel, et à son marie Djillali, à qui je dois ce que je suis maintenant.

A ma sœur Zoubida et son marie Fauzi pour leur compréhension.

A mon petit frère Fethi, mon supporteur fidèle.

A ma sœur Razika, pour sa gentillesse.

A mon grand frére Mohamed, qui me manque beaucoup.

A toute la famille Hamlati.

A la très chère Nadia qui m'a offert le plus beau cadeau du monde, et à sa sœur « WARDA » qui n'a pas cessé de prier pour mois, merci warda.

A maamar mon meilleur amis.

A mon binôme Hanane, pour sa patience, et à sa famille (père mère et frère).

A mon ami Hicham qui à était toujours à coté de mois et qui m'a aidé dans des moments très difficiles.

A mes très chère amis (es), qui m'ont toujours encouragées, surtout Halim el-petit, Chakir, yazid, sidaali, Nacer, Houbaibe, Hamou et son frère Mounir, …

A toute la promo, pour l'affectueuse collaboration durant toute les 5 années.

Et à tous les gens qui m'ont aidé à élaborer ce travail de prés ou de loin, surtout les gent de BLIDA.

Anis

Dédicaces

A ma chère mère qui, avec des yeux pétillants d'amour m'inculqua la valeur du travail bien accompli.

A mon chère père qui plus de quiconque, me prouva l'importance qu'il fallait attacher aux sentiments des autres.

A mon unique frère Maamar

A ma chère grand mère maternelle

A ma chère tante paternelle

A toute ma famille

A tous mes amis (es)

Je dédie ce mémoire

Hanane

Remerciments

Notre sincère reconnaissance à notre promoteur M^{r} OULD BRAHEM, consultant Microtel auprès de Sonatrach pour sa disponibilité constante et l'intérêt avec lequel il a suivi notre travail.

Nous tenons à remercier notre co-promotrice M^{lle} DJAOUDI ingénieur en informatique à la Sonatrach pour nous avoir guidé et conseillé.

Nos vifs remerciment à M^{r} BENOUAR, M^{r} BOUKHELF et M^{r} OULD AISSA de la commission de suivie pour nous avoir fait l'honneur de nous encadrer.

RESUME

A ses débuts, la communication entre ordinateurs étaient une véritable aventure.
De nos jours, il existe des milliers de réseaux locaux hétérogènes éparpillés dans le monde entier, permettant la communication et l'échange des services entre des équipements distants. Pour garantir la fiabilité de l'information échangée et la meilleure exploitation des ressources du réseau, une bonne gestion de ces dernières est indispensable.

Notre objectif principal est de réaliser un logiciel qui permet aux administrateurs de réseau d'avoir, d'une part une vue globale sur le réseau, et sur les protocoles transitant sur ce réseau, et d'autre part de fournir une interface graphique conviviale permettant aux administrateurs :

- ❖ D'être informé sur les anomalies (alertes) signalées dans le réseau.
- ❖ D'introduire (modifier) la qualité de service selon les besoins.

Notre travail consiste à développer une application qui permet d'extraire des informations telles que les noms des postes en utilisant le standard SNMP (Simple Network Management Protocol), de récolter les paquets transitant sur le réseau à un moment donné pour la collecte des informations concernant les protocoles, de les analyser et d'extraire des informations à partir des statistiques obtenues afin de déceler les goulots d'étranglement, d'améliorer l'utilisation de la bande passante et la qualité de service sur l'ensemble des éléments constituants le réseau.

Mots clés :

- Le protocole de gestion SNMP (Simple Network Management Protocol),
- La MIB (Management Information Base),
- L'agent SNMP,
- La station de gestion NMS
- La qualité de service QoS

Table des matières

Chapitre II : Le protocole TCP/IP

Chapitre III : Le protocole SNMP

Chapitre IV : Etude conceptuelle

Chapitre V : Mise en œuvre

Conclusion générale

Bibliographie

Liste des figures

Chapitre I : Le modèle de référence OSI

Chapitre II : Le protocole TCP/IP

Chapitre III : Le protocole SNMP

Chapitre IV : Etude conceptuelle

Chapitre V : Mise en œuvre

Liste des tableaux

Chapitre I : Le modèle de référence OSI

Chapitre II : Le protocole TCP/IP

Chapitre III : Le protocole SNMP

Liste des acronymes

API	Application Programming Interface
ARP	Address Resolution Protocol
ARQ	Automatic Repeat reQuest
ASN1	Abstract Syntax Notation One
BER	Basic Encoding Rules
BGP	Border Gateway Protocol
BPF	Packet Capture Driver
CCITT	Comité Consultatif du Télégraphe et du Téléphone
CEI	Commission Eléctrotéchnique Internationale
COPS	Common Open Policy Services
CSMA/CD	Carrier Sense Multiple Access with Collision Detect
DEN	Directory ENAbled Networking
DES	Data Encryption Standard
DiffServ	Differentiated Service
DLL	Dynamic Link Library
EGP	Exterior Gateway Protocol
FIFO	First In First Out
FTP	File Transfer Protocol
HTTP	Hyper Text Transfer Protocol
HUB	Host Unit Broadcast
IAB	Internet Activities Board
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
IGP	Interior Gateway Protocol
IntServ	Integrated Service
IOCTL	Input Output Control
IP	Internet Protocol
IPX	Internet work Packet EXchange
ISO	International Standard Organisation
ISP	fournisseur d'accès à Internet
ITU	International Telecommunication Union
JTC	Joint Technical Committee
LAN	Local Area Network
Libpcap	Library Packet Capture

MAC	Media Access Control
MAN	Metropolitan Area Network
MD5	Message Digest Version 5
MFC	Microsoft Fondation Class
MIB	Management Information Base
MN	Management Node
MTU	Maximal Taille Unit
NCP	Netware Core Protocol
NDIS	Network Driver Interfacez Specification
NIC	Network Interface Card (Network Information Center)
NMS	Network Management Station
OID	Object Identifier
OSI	Open System Interconnection
PDU	Protocol Data Unit
PER	Packet Encoding Rules
QoS	Quality of Service
RFC	Request For Comments
RIP	Routing Information Protocol
RMON	Remote network MONitoring
RSVP	Resource Reservation Protocol
SGMP	Simple Gateway Monitoring Protocol
SHA-1	Standard Hashing Algorithm Version 1
SMI	System Information Base
SMTP	Simple Mail Transfert Protocol
SNMP	Simple Network Management Protocol
SPX	Sequenced Packet Exchange
TCP	Transport Control Protocol
TELNET	TELecommunication NETwork
UDP	User Datagramme Protocol
UIT-T	Union Internationale des Télécommunications-secteur de normalisation des Télécommunications
USM	User Security Model
VACM	View Access Control Model
WAN	Wide Area Network
WFQ	Weighted Fair Queuing
WinSockAPI	Windows Sockets Application Programming Interface

Introduction générale

Introduction générale

L'informatique a marquée sa présence dans différents domaines en évoluant.
Les réseaux informatiques touchent de plus en plus notre quotidien, leur croissance a été très rapide, le contrôle de leurs topologies est devenu très difficile et les besoins en gestion ont beaucoup augmentés ces dernières années.

La diversité des équipements (fabricants, modèles…) et le besoin d'interconnexion ont crées des outils spécialisés pour gérer ces réseaux.
Donc il y a nécessité de surveiller et contrôler à distance les réseaux et agir en cas de nécessité pour rendre des services meilleurs et convenables en récoltant les données de gestion des équipements de réseaux, en utilisant des protocoles de gestion de réseaux.

Il existe deux façons de gérer les réseaux :

- Gestion des réseaux d'entreprise : l'outil est basé sur SNMP
- Gestion des réseaux des opérateurs (ATN) : outils CMIP et Q3, très complexe et très lourd aux déploiements.

Nous allons consacrer notre projet sur la gestion des réseaux d'entreprise basés sur SNMP.

Nous proposons une application qui permet de surveiller un réseau à distance en utilisant le protocole de gestion de réseau SNMP : Simple Network Managment Protocol, traduisez protocole simple de gestion de réseau ; pour pallier aux besoins de gestion et pour disposer d'une cartographie du réseau, fournir un inventaire précis de chaque machine, capter le trafic à des fins d'analyse, et faire des statistiques pour mesurer la consommation d'une application, et signaler les dysfonctionnements.

SNMP est un protocole très simple et facile à utiliser, il permet une gestion à distance des différentes machines, le modèle fonctionnel pour la surveillance et pour la gestion est extensible, il est indépendant de l'architecture des machines administrées.

La première version de ce protocole est dérivée du protocole SGMP (Simple Geteway Monitoring Protocol), et a été crée par l'IETF (Internet Engenneering Task Force) en 1988, dans le but, d'administrer des appareils connectés à un réseau, d'identifier les différents composants d'un appareil (interfaces réseau, logiciels installés, tables de routage, …), d'obtenir des mesures des différents flux passant par l'appareil (nombre de trames, nombre de paquets IP entrant, sortant, fragmentés, …), de changer les paramètres de fonctionnement, et fédérer en un standard unique des protocoles multiples liées chacun à un type d'équipement ou à une marque pour centraliser et simplifier la gestion des réseaux.
Le standard SNMP est basé sur une entité gérée « Agent » et une entité de gestion « Manager » et un protocole de gestion qui sert à transporter les informations entre ces entités.

Pendant la dernière décennie, les besoins en gestion de réseaux ont beaucoup augmentés. Certaines organisations ont plusieurs milliers d'unités à gérer dans plusieurs domaines de l'administration, aussi la gestion d'équipements se fait maintenant par les réseaux publics et privés. Comme la première version de SNMP ne comporte pas de mécanisme de sécurité, elle ne pouvait pas être utilisée sur les réseaux publics.

La version 2 est apparue pour pallier aux problèmes de la première version, elle est plus complexe et contient un niveau hiérarchique d'administration (dialogue gestionnaire à gestionnaire : l'entité SNMP joue le rôle d'agent ou de gestionnaire suivant les tâches qu'on lui demande), elle incorpore un niveau de sécurité, mais il ne s'agit toujours pas d'un standard complet.

SNMPv2 ne répond pas à tous les besoins de gestion de réseau et ne résout pas le problème de sécurité.

Alors une nouvelle version SNMPv3 a vu le jour, elle est très avancée dans son cheminement vers le standard, elle est décrite en sept documents. Elle vise à inclure la sécurité des échanges entre la station de gestion et les éléments gérés.

La configuration doit être cohérente pour tous les agents SNMP dans un réseau donné.

Notre application consiste à implémenter la cartographie du réseau, ceci consiste à récolter les informations existantes dans les MIBs de chaque élément du réseau (c'est la partie SNMP), établir un sniffer qui permet de récolter et capter les paquets transitant sur le réseau en écoutant et scrutant le réseau, cette partie utilise une méthode de conception pour réaliser la capture des trames, cette méthode se décompose en trois niveaux :
Le niveau1 : Le Noyau de l'architecture, c'est le module qui réalise effectivement la capture des trames
Le niveau2 : Le Module d'interface et de Dialogue fait l'interface de communication entre le Niveau1 (niveau noyau) qui réalise la capture, et le niveau3 (niveau application de l'utilisateur) qui attend la remontée des paquets de données capturés.
Le niveau3 : Le Module de capture utilisateur, c'est le programme de capture de l'utilisateur qui va faire des traitements sur les paquets capturés.
Après avoir fait remonter et récolter les paquets transitant sur le réseau, il faut faire la classification du trafic et statistiques, ceci consiste à déterminer les protocoles circulants sur le réseau, et les classés selon l'occupation de la bande passante, et ensuite d'utiliser ces statistiques pour configurer et mettre en place, une qualité de service, par exemple, c'est à dire établir des priorités sur les paquets selon les besoins de l'utilisateur.

Dans ce mémoire, nous allons présenter les différentes étapes suivies, pour la réalisation de notre projet, et qui sont exposées dans les chapitres suivants :

- Chapitre1 : portera sur les systèmes ouverts, et particulièrement sur une étude approfondie du modèle de référence OSI comme standard.
- Chapitre2 : traitera d'une façon détaillée le modèle le plus répondus TCP/IP.
- Chapitre3 : abordera de façon claire et approfondie les trois versions du protocole de gestion SNMP et particulièrement le protocole SNMPv3 et la manière dont il assure la sécurité sur le réseau.
- Chapitre4 : décrira l'approche conceptuelle de notre système.
- Chapitre5 : développera la réalisation de notre approche suivie par des tests et résultats.

Chapitre I : Modèle de référence OSI (Open System Interconnection)

I- Introduction

En 1977 l'Organisation Internationale de Normalisation (ISO) a créé pour des besoins de compatibilités entres les différentes machines tout un ensemble de lois de compatibilités en différentes couches baptisé modèle OSI (Open System Interconnection model). Celui ci est appelé modèle de référence OSI parce qu'il traite de la connexion entre les systèmes ouverts, c'est à dire des systèmes ouverts avec d'autres systèmes.
Le modèle OSI préconise le découpage de la communication en 7 couches, afin de permettre de normaliser les méthodes d'échange entre deux systèmes.
Chaque couche a un rôle bien particulier et communique sur requête (sur demande) de la couche supérieure en utilisant les services de la couche inférieure (sauf pour la première couche : physique).
Une couche ne correspond pas forcement à la spécification d'un seul protocole, mais elle définit une fonction de transmission de données qui peut être exécutée par un nombre quelconque de protocoles.

Dans la section suivante, nous retrouverons les systèmes fermés et puis les systèmes ouverts.
La section 4, portera sur le concept d'architecture en couche. Ensuite, la section 5, introduira la structure du modèle de référence OSI.

II- Définition des systèmes fermés

Dans un groupe d'ordinateurs fermés seules les machines de même type peuvent communiquer entre elles.

III- Définition des systèmes ouverts

Un système ouvert permet aux utilisateurs d'ordinateurs de marque quelconque de communiquer avec d'autres marques s'ils sont tous les deux Conformes aux standards internationaux.
Pour faciliter l'introduction des systèmes ouverts, il faut que les standards soient reconnus à une échelle internationale (**I**nterconnexion des **S**ystèmes **O**uverts 'ISO').
Chaque pays possède des organismes de standardisation couvrant différents domaines ils se regroupent au niveau international et se mettent d'accord sur des standards internationaux.
L'ITU(union internationale des télécommunications)est un de ces organismes.

IV- Le concept d'architecture en couche

Le concept d'architecture en couche demande la définition de trois objets de niveau (N), qui eux-mêmes seront déterminés par des attributs : la sémantique d'association, la sémantique de fonctionnalité et la syntaxe du codage. Le niveau (N) est composé de trois objets.

- Le service (N).
- Le protocole (N).
- Les points d'accès au service (N)-SAP ((N)-**S**ervice **A**ccess **P**oint).

1- Le service (N)

C'est le service qui doit être rendu à ce niveau de l'architecture, il doit être réalisé par un ensemble de taches devant être effectuées à la couche N pour rendre un service à la couche située au-dessus d'elle. Le service N correspond aux évènements et aux primitives associées à mettre en place, pour rendre un service au niveau supérieur.

2- Le protocole (N)

C'est un ensemble de règles qui définissent les mécanismes nécessaires pour que le service de niveau (N) soit réalisé et pour contrôler l'envoi des données, d'un niveau (N) à un autre niveau (N) correspondant au service (N).

3- Les (N)-SAP ((N)-Service Access Point)

Ils sont situés à la frontière entre les couches (N+1) et (N).
Les différents paramètres pour la réalisation du service (N) s'échangent sur cette frontière.
Un (N)-SAP permet d'identifier une entité de niveau (N+1). C'est à ces points frontières que les adresses sont définies.

Un service et un protocole sont déterminés par trois attributs :

- La sémantique d'association.
- La sémantique de fonctionnalité.
- La syntaxe.

La sémantique d'association c'est le mode de transmission des données, il peut être soit orienté connexion soit non orienté connexion.
Les différentes phases de la communication sont caractérisées par l'échange de :

- Primitive de service.
- D'unité de données de protocole ou PDU (**Protocol Data Unit**).

La sémantique de fonctionnalité regroupe l'ensemble des procédures qui seront utilisées pendant la phase de transfert de données.
Pour une transmission avec connexion les principales fonctions que nous rencontrerons sont :

- Fragmentation/ré assemblage.
- Concaténation/séparation.
- Données exprès.
- Remise en séquence.
- Réinitialisation.
- Contrôle de flux.
- Contrôle d'erreurs.

La syntaxe caractérise le codage des primitives de service et des PDUs (**Protocol Data Unit**).

4- Les entités

On appel entités les éléments actifs de chaque couche. Elle peut être une entité logicielle (comme un processus) ou une entité matérielle(comme une puce d'E/S intelligente). Les entités de la même couche sur différentes machines sont appelées **entités paires**. Les entités de la couche 7 sont appelées **entités d'application**, les entités de la couche 6 **entités de présentation** et ainsi de suite.
Les entités de la couche N activent un service utilisé par la couche N+1. La couche N est alors appelée fournisseur de service et la couche N+1 utilisateur de service. La couche N est utilisateur de services de la couche N-1. Elle peut offrir différentes catégories de service, par exemple des services de communication rapides et chers ou au contraire lents et économiques.
Les services sont accessibles par des points d'accès aux services, **SAP**(**Service Access Point**).

5- Les unités de données

Les principales unités de données sont :

- **(N)-SDU** ((N)-Service Data Unit) : unité de données du service (N): est un ensemble de données provenant de l'interface avec la couche (N) qui doit être transporté sur une connexion (N).
- **(N)-PCI** ((N)-Protocol Control Information) : les informations de contrôle du protocole (N) proviennent d'entités (N) pour coordonner leur travail. Elles sont rajoutées à des SDU sur une connexion (N-1).
- **(N)-PDU** ((N)-Protocol Data Unit) : les unités de données de protocole (N) sont spécifiées par un protocole (N), et consistent en des informations de contrôle du niveau (N) et d'informations provenant d'une unité (ou plusieurs) de données de service.
- **(N)-ICI** ((N)-Interface Control Information) : permet de transportée les informations nécessaires (de gestion) pour contrôler la communication entre entités de niveau (N+1) et entités de niveau(N).
- **(N)-IDU** ((N)-Interface Data Unit) : regroupe toute les information de gestion qui sont ajoutées aux données à transporter au travers de l'interface (N) ((N)-PDU). **[Puj 98]**

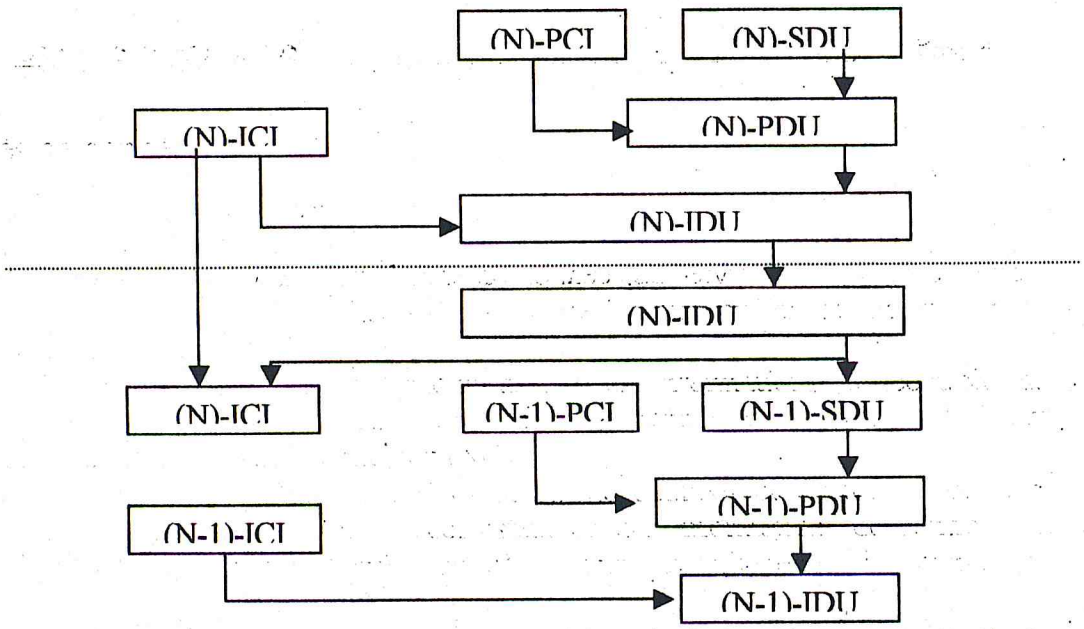


Figure 1 : Une concaténation de niveau (N) suivie d'une segmentation de niveau (N-1)

On peut appeler ces données à transporter au travers de l'interface des données de l'interface (N) ou (N)-ID ; elles proviennent des données utilisateurs (N) ou (N)-UD.
Jusqu'ici nous avons parlé que de la couche (N). Pour parler de chaque couche séparément nous utilisons une lettre correspondant à chaque niveau. Par exemple pour la couche application : ASDU, APCI, APDU, AICI, AIDU.

6- Les primitives de service

Un service est défini formellement par un ensemble de primitives (opérations) qui disent au service de réaliser telle action ou de rendre compte d'une action prise par l'entité paire. Dans le modèle OSI les primitives de service peuvent être réparties en quatre classes :

Primitive	Signification
Request (demande)	Requêtes : une entité sollicite un service pour faire une activité, par la quelle un utilisateur de service appelle une procédure.
Indication	Indication : une entité est informée qu'une procédure (événement) est mise en route par l'entité émettrice sur son SAP ou bien qu'un fournisseur de service indique qu'il appelle une procédure.
Responce	Réponse : une entité répond à un événement.
Confirmation	Confirmation : une entité est informée de sa demande de service.

Tableau 1 : Les classes des primitives de service

La primitive de **demande** ordonne un travail, par exemple établir une connexion. Quand le travail est exécuté, l'entité paire reçoit une primitive d'**indication** annonçant que quelqu'un veut établir une connexion avec elle, alors elle envoie une primitive de **responce**, pour dire si elle accepte ou rejette la proposition de connexion. Dans tous les cas, l'entité émettrice de la requête initiale est avertie de ce qui s'est passé via la primitive **confirmation**.

Les paramètres des primitives :

Les primitives peuvent avoir des paramètres :

Les paramètres d'une primitive de **demande** de connexion spécifient la machine à laquelle on désire se connecter, le type de service désiré et la taille maximale des messages utilisée pour la connexion.

Les paramètres d'une primitive **indication** peuvent contenir l'identité de l'appelant, le type de service désiré et la taille maximale des messages proposée.

Les paramètres d'une primitive **response** peuvent contenir des contre-propositions.

Les paramètres de primitive **confirmation** peuvent contenir les contre propositions. **[Puj 98]**

V- Structure de l'OSI

L'OSI est composée de sept couches, les trios premières (physique, Liaison de données, réseau) s'appellent couches orientées réseaux, leur rôle est l'établissement des connexions réseaux et la transmission fiable des données sur la liaison.

Les trois dernières (session, présentation, application) s'appellent couches orientées utilisateurs, elles s'occupent de l'établissement de la connexion et du transfert de donnée au nom de l'utilisateur de l'application. **[Tau 99]**

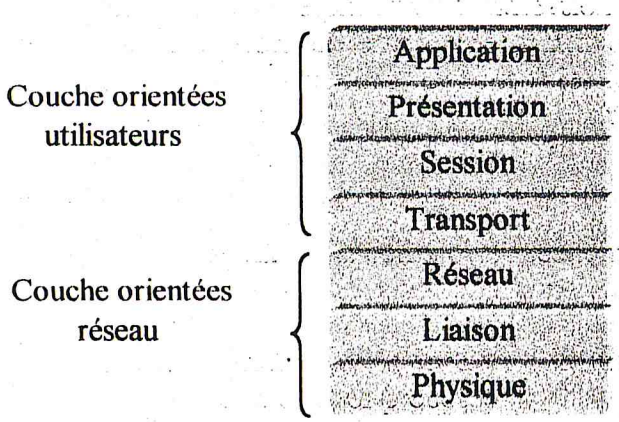


Figure 2 : L'architecture OSI

La 4emme couche relie les couches orientées utilisateurs et les couches orientées réseaux, elle fournit un service de transmission fiable aux couches utilisateur indépendamment du réseau utilisé, on peut la considérer comme la charnière du modèle.

1- Acheminement (Passage) des données entre les couches

Aucune donnée n'est transférée directement de la couche N d'une machine à la couche N d'une autre machine pour cela on distingue deux types de communications entre couche :

1. Chaque couche passe les données et le contrôle à la couche immédiatement inférieur, jusqu'à la plus basse en dessous de la couche une, oû se trouve le support physique, et puis le transport des messages (des données) se fait par l'intermédiaire de sous-réseau. Chaque couche ajoute un en-tête est passe les données à la couche inférieure. A l'arrivée les différents en-têtes sont éliminés un par un lorsque le message remonte dans les couches jusqu'à parvenir au processus récepteur. On parle d'une communication **verticale. [Joe, Dan 97]**

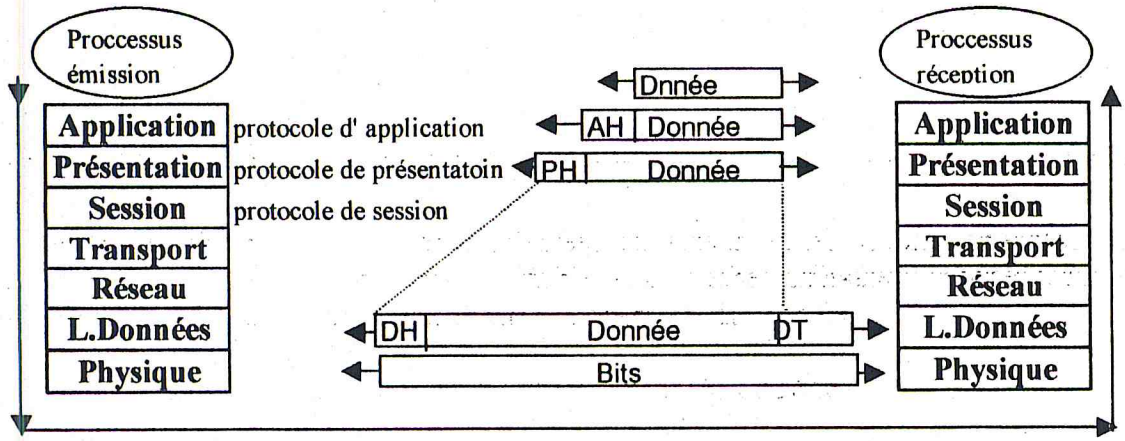


Figure 3 : Acheminement des données entre les couches de OSI

Entre chaque paire de couche adjacente, on trouve une interface. L'interface définit les opérations élémentaires et les services que la couche inférieure offre à la couche supérieure.

2. Les transmissions entre les couches homologues (paire) se font via des protocoles. Chaque protocole ne tient pas compte de la couche située au-dessus de lui. On parle alors d'une communication **horizontale.**

VI- La couche Physique - Physical Layer (C1)

La couche Physique établit la connexion physique entre le système OSI et le réseau physique, elle permet de véhiculer l'information et de transformer des séquences de bits (0 ou 1) en séquence de grandeur physique appropriée au médium de communication (codage et décodage des signaux par le modem ou autre interface de connexion comme les 'jonctions', multiplexeurs, les nœuds de commutation...).
Elle contient toutes les spécifications concernant la transmission de données sur un réseau physique, quel que soit son type. Parmi ces spécifications :

-La vitesse de transfert des données (ou vitesse d'horloge).
-Le type de câble utilisé (coaxial, paire torsadée ou fibre optique).

-Le type de connecteurs (BNC, RJ45) et le nombre de broches utilisées.
-La topologie physique.
-Le niveau du signal électronique ou lumineux représenté par un bit (1 ou 0).

Elle est chargée des fonctions suivantes :

1- La fonction mécanique
Elle décrit la forme des connecteurs ainsi que le type et le nombre de fils ou de longueur d'onde.

2- La fonction de transmission

Elle définit les caractéristiques physiques du signal. Si le médium est un câble métallique, il s'agit des problèmes suivants:

- Le nombre de volts à atteindre pour représenter un bit à 1et à 0.
- La durée d'un bit en microsecondes.
- La possibilité de transmission dans les deux directions simultanément.
- Le nombre de broches que possède le connecteur de réseau et l'usage de chacune.

3- La fonction fonctionnelle

Elle définit les méthodes de transmission de données (flux, synchronisation).

4- La fonction procédurale

Elle est chargée d'établir et de maintenir la liaison avec une machine distante.

Les périphériques utilisés par la couche physique :
Les types de périphériques, le plus souvent rencontrés à ce niveau, lors d'une connexion de réseaux sont :

- Hub.
- Répéteur.
- Commutateur
- Modem

VII- La couche Liaison de données - Data Link Layer (C2)

Elle est responsable de la transmission des données sur la liaison physique sous-jacente en assurant la meilleure qualité de transmission possible. Elle l'accomplit comme suit :

- Eclate le message d'entrée de l'émetteur en morceau : **trames**(de quelques centaines d'octets).
- Transmet des trames en séquence et en gérant les trames d'acquittement renvoyées par le récepteur, c'est à dire la retransmission des trames autant de fois qu'il est nécessaire.

Tandis que la couche physique accepte et transmet que des flots de bits sans connaître la signification ni la structure, la CLD crée et reconnaît les frontières des trames. Cela peut se réaliser en accolant au début et à la fin de la trame un délimiteur de début et de fin de trame (en-tête & en-queue).

Remarque :
Si de tels groupes peuvent apparaître accidentellement dans les données, il convient de prendre des précautions spécifiques pour les confusions.

Elle assure aussi le contrôle des accès vers l'interface réseau, assemble une trame de donnée puis lui ajoute les informations d'adressage. Les protocoles Ethernet (IEE802.3), Token Ring (IEE802.5), PPP, SLIP sont des standards qui travaillent sur cette couche.
En masquant aux autres couches les différences physiques du réseau. Elle assure le séquansement des trames (numérotation des trames).

Elle assemble les données en blocs, auxquels elle ajoute des informations de contrôle pour constituer une trame de données : l'adresse de destination, la longueur du message, l'information de synchronisation, de détection d'erreur, etc.

Une autre fonction de cette couche est la gestion de flux, c'est-à-dire empêcher un émetteur rapide d'inonder de données un récepteur lent, une sorte de mécanisme de régulation de trafic.

Les protocoles qui fonctionnent à ce niveau, délivrent des données de carte à carte (l'adressage physique)

La couche Liaison de Données se découpe en deux sous-couches :
1- Logical Link Control (LLC).
2- Medium Access Control (MAC).

Les caractéristiques de la couche liaison de données dépendent :
- Du mode d'exploitation de la ligne : duplex, full duplex, simplex.
- Du mode de détection d'erreurs.
- Du format des trames.
- Du principe d'acquittement utilisé (acquittement : ack).

1- Les protocoles de la couche

Des exemples WAN de ces types de protocoles incluent :
- High Level Datalink Control (HDLC), LAP B utilisé par des réseaux X.25
- PPP
- Frame Relay
- ATM

Dans les réseaux LAN, il existe plusieurs protocoles. Les plus importants sont ceux de la série 802 qui décrivent les méthodes d'accès aux câbles, les formats de trame et l'adressage physique. Ces protocoles sont les suivants :
- Ethernet (802.3)
- Token Bus (802.4)
- Token Ring (802.5) **[Joe, Dan 97]**

2- Les règles appliquées par cette couche

1- l'acquittement des trames.
2- le contrôle de flux.
3- le contrôle d'erreur.

2-1- L'acquittement des trames

La CLD reçoit un paquet de donnée que lui transmet la couche réseau.
La CLD offre des services selon des cas :
1- Service sans connexion.
2- Service avec acquittement sans connexion.

3- Service orienté connexion.

Avec le service **sans connexion,** il n'est pas nécessaire d'établir une connexion pour commencer la transmission de donnée (la trame est transférée de l'émetteur vers le récepteur sans confirmation d'acceptation).
Avec le **service avec acquittement sans connexion,** elle exige l'acquittement des trames, les acquittements doivent arriver continuellement pour permettre une circulation efficace des données. Ces deux services sont utilisés généralement dans les réseaux locaux.
Le **service orienté connexion** s'inspire du réseau téléphonique il faut établir un appel avant de transférer les informations, les étapes se font dans l'ordre suivant :

- Etablire un appel (établire une connexion).
- Transférer les données.
- Libérer l'appel (libérer la connexion).

2-2- Le contrôle de flux

Le contrôle de flux est très important pour le transfert de donnée sans erreur, lorsqu'une extrémité réceptrice ne peut plus accepter des données (la mémoire tampon est complète) alors que l'extrémité émettrice continue à émettre, dans ce cas il y à trois problèmes :

1- Toutes les données ne peuvent être exécutées.
2- Une partie des données peut être perdue.
3- Un ordinateur peut exiger une transmission continue des données et créer un encombrement.

La solution consiste à empêcher les ordinateurs (émetteurs) d'envoyer des données tant que l'extrémité réceptrice n'est pas prête.
Il existe plusieurs mécanismes permettant la circulation efficace des données parmi ces mécanismes :
Contrôle de flux X-ON/X-OFF :
L'extrémité réceptrice envoi un caractère spécial **X-OFF** à l'extrémité émettrice pour arrêter d'envoyer, lorsque l'extrémité réceptrice est prête à recevoir des données, elle envoi le caractère **X-ON** à toutes les extrémités émettrices.
Mécanisme de freinage :
Ce mécanisme permet de transférer un certain nombre de trames à tout moment. Il peut y avoir un nombre maximum de trames non acquittées selon la taille de la fenêtre utilisée.
Dans l'en-tête standard, le champ de séquence fait 3 bits de long et sert à numéroter les trames et à les acquitter, comme il y a 3 bits, leur valeur maximum est 7.
L'extrémité réceptrice envoi des ACK pour acquitter toutes les trames sans erreur.

2-3- Contrôle d'erreur

Consiste à détecter les trames altérées et à demander leur retransmission, lorsqu'une erreur est détectée, la reprise s'effectue en envoyant ARQ (Automatic Repeat reQuest).
Il existe deux types d'ARQ :

1- ARQ arré-attente.
2- ARQ continue.

ARQ arré-attente :
L'ARQ arré-attente utilise le mécanisme de freinage, la taille maximale est réduite à 1.
Il y a 3 contraintes :
- Recevoir un ACK par l'émetteur pour transmettre d'autre.
- Expiration d'un temporisateur et retransmission de la trame.
- Réception d'un NACK (un acquittement négatif c.à.d la trame a été reçue mais pas correcte).

- Il n'y a qu'une trame en suspens à la fois.
- A cause de la valeur 1 seule une copie de la trame doit être conservée dans la liste de transmissions.

Les inconvénients :
- Lenteur du flux de données puisque l'on ne transmet qu'une seule trame à la fois.
- Le problème de duplication de trame se produit en cas de perte d'ACK.

Un des moyens de supprimer le second inconvénients consiste à étiqueter chaque trame de 0 ou 1 alternativement pour identifier plus facilement la trame acquittée.

ARQ continue :

Il existe deux variantes d'ARQ :

1- ARQ retour-x
2- ARQ sélective

- **ARQ retour-x :**

De multiples trames sont envoyées et acquittées par la destination jusqu'à ce que la source reçoive une trame NACK.
Le signal retour-X indique que l'extrémité émettrice doit revenir à la trame manquante et reprendre la transmission à partir de ce point même si elle a envoyé d'autres trames avant la réception du NACK.

L'inconvénient :

L'inconvénient de cette méthode est que l'extrémité émettrice doit conserver une copie de chaque trame non acquittée en suspens ce qui exige beaucoup de mémoire tampon.

L'avantage :

L'avantage est qu'il est facile à implémenter dans la mesure ou la séquence de trames envoyées reste la même.

- **ARQ sélective :**

Avec ARQ sélective seule les trames hors séquence sont retransmises.
Exemple : si la station 'A' envoi trois trames, la station 'B' reçoit la première et la troisième, elle envoi alors NACK à la station 'A'. A la réception de NACK, la station 'A' envoi alors la 2^{emme} trame à B.

L'inconvenant :

L'inconvénient est quelle exige un large volume de mémoire tampon, ainsi que du temps système pour reclasser les trames hors séquence.
L'extrémité émettrice doit également être en mesure de renvoyer les trames hors séquence.

3- Le niveau Ethernet

Son principe est basé sur la diffusion des messages sur un bus logique, qui peut être un bus ou une étoile physique, où tous les hôtes partagent de façon équitable le support, elle envoie une trame(**trame Ethernet**) contenant le message, sa propre adresse et l'adresse du destinataire sur le support physique.
Toutes les stations voient passer cette trame, mais seule la station concernée la lit. La communication se fait à l'aide d'un protocole appelé CSMA/CD (Carrier Sense Multiple Access with Collision Detect) ce qui signifie qu'il s'agit d'un protocole d'accès multiple avec surveillance de porteuse (Carrier Sense) et détection de collision.

L'architecture Ethernet est constituée de deux couches fondamentales : la couche physique et la couche de contrôle. Ces deux couches correspondent respectivement aux couches 1 et 2 du modèle OSI.

Les réseaux Ethernet se nomment 10Base5, 10Base2, 10BaseT, 10Broad36,etc. Ce sont des notations IEEE, elles dépendent du débit utilisé, du mode de transmission et du câblage utilisé.

Les câbles utilisés sont :
- Gros coax
- Coax fin (cheapernet ou Ethernet fin)
- UTP (unshielded twisted pair) 3 et 5, c'est à dire paire torsadée téléphonique et paire torsadée de meilleure qualité
- Fibre Optique MMF (Multi Mode Fiber) et SMF (Single Mode Fiber)

La distance maximale entre deux répéteurs dépend de l'atténuation du signal et donc de la qualité du support.

3-1- Trame Ethernet

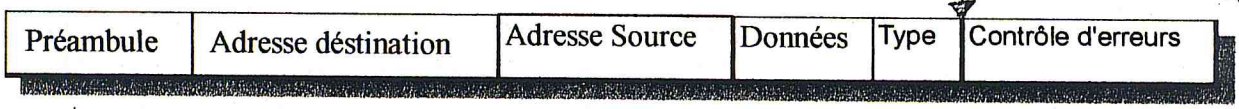


Figure 4 : Trame Ethernet

Préambule : pour délimiter une trame (montre le début d'une trame)
Adresse destination : sur 6 octets représente l'adresse physique (Ethernet) de la station devant recevoir la trame.
Adresse source : sur 6 octets représente l'adresse physique Ethernet de la station ayant émis la trame, de taille identique au champ d'adresse de destination.
Type : sur 2 octetes représente le type de données transmises selon que c'est un datagramme IP, une requête ou réponse ARP ou RARP, longueur de la trame exprimée en octets.
Données : de 46 à1500 octets (données à transmettre).
CRC : sur 4 octets

VIII- La couche Réseau - Network Layer (C3)

La couche réseau permet aux couches supérieures d'être indépendantes des différents types de liaisons de données ou technologies de transmissions.
Elle doit connaître la topologie du réseau de communication pour déterminer le meilleur itinéraire pour l'envoie d'un message à travers celle – ci.

Parmi les principales fonctions de la couche réseau ; le contrôle de congestion.
Quand un paquet doit transiter d'un réseau à un autre pour arriver à destination, beaucoup de problèmes peuvent surgir.
- L'adressage dans le second réseau peut être différent de celui du premier.
- Le second peut ne pas accepter du tout le paquet à cause de la voluminosité de ce dernier.
- Les protocoles peuvent varier.

C'est à la couche Réseau de résoudre tous ces problèmes pour permettre l'interconnexion de réseau hétérogène.
Elle s'occupe de l'adressage logique et du routage des paquets à leurs destinations ainsi que le contrôle de flux.

1- La fonction routage

Permet d'aiguiller les paquets de données depuis un système A vers un système B. Les paquets peuvent emprunter plusieurs chemins selon l'encombrement du réseau. Chaque paquet transmis comporte l'adresse de la machine de destination. Quel que soit le chemin emprunté, la donnée sera transmise au système B.

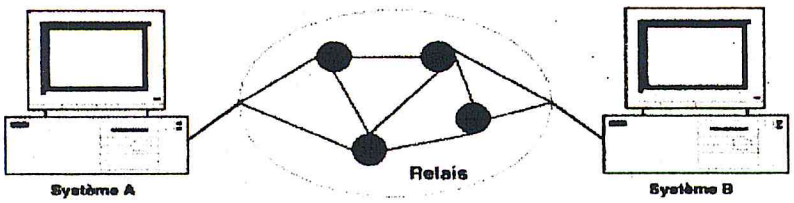


Figure 5 : La communication entre deux systèmes à travers un relais

Les algorithmes de routage doivent être :

-exacts
-simples
-robustes
-optimaux
-stables

2- La fonction gestion de congestion

Elle permet de résoudre les problèmes d'encombrement rencontrés en cours de route. Si un relais (ensemble de nœuds) est saturé, le paquet sera dévié vers un autre nœud.

3- Les modes de connexion

Deux modes de connexion sont possibles :
-Le mode connecté.
-Le mode non connecté.

4- Les protocoles utilisés pour cette couche

Les protocoles suivants sont actuellement utilisés pour cette couche :
Internetwork Packet Exchange (IPX) de Novell
Internet Protocol (IP)
X.25
Les protocoles de routage interviennent à ce niveau tels **R**outing **I**nformation **P**rotocol (RIP).

Note :

Les adresses de réseau sont composées de deux parties :

1- Code d'identification du réseau de donnée DNIC.
2- Adresse du réseau NUA.

Remarque:

Chaque réseau de commutation par paquet a un code DNIC unique.
Il est composé de quatre chiffres (DNIC), les trois premiers se rapportent au pays de résidence de l'utilisateur, le 4^{emme} indique le réseau par paquet au quel l'utilisateur est connecté à ce pays.
L'adresse utilisateur comporte dix chiffres.
Remarque : DNIC pour les réseaux internationaux.

IX- La couche Transport - Transport Layer (C4)

La couche transport s 'appuie sur une méthode d'adressage indépendante des conventions utilisées dans les couches inférieures. Son rôle sera de réaliser une correspondance entre l'adresse de transport d'un utilisateur donné et l'adresse de réseau correspondante, pour pouvoir initialiser la communication.
Son rôle également est de faire le lien entre les services offerts par la couche réseau, et les besoins de l'utilisateur. Elle isole les niveaux supérieurs de la technologie de la couche réseau, en proposant un service standardisé. Ainsi, un changement dans la couche réseau ou un changement dans la technologie du matériel n'entraînera aucun changement dans les programmes situés dans les couches supérieurs.
Elle est chargée d'établir les connexions de bout en bout de l'émetteur au destinataire, elle est chargée aussi de maintenir la qualité de la connexion et de la libérer de manière ordonnée, une fois la conversation terminée.

La couche transport accepte des données de la couche session, les segmente en petites unités, et s'assure que tous les morceaux arrivent correctement de l'autre côté.

La couche transport peut créer de multiples connexions réseau sur lesquels elle répartit les données pour améliorer le débit, cela se fait, que si la connexion de transport requiert un débit rapide de la couche session.

Dans le cas où la création et le maintien d'une connexion seraient coûteuses, la couche transport peut multiplexer plusieurs connexions transport sur la même connexion réseau (pour réduire le coût). Le rôle de la couche transport est de rendre ce multiplexage transparent à la couche session.

1- Les fonctions de la couche

- Optimiser les ressources réseaux.
- Dépendance du service réseaux.
- Elle segmente en blocs les données qu'elle reçoit de la couche session pour créer des blocs, que la couche réseau pourra accepter (bloc de la taille 128 ou 256 caractères).
- Elle utilise les numéros de séquence des extrémités source et destination.
- Elle détecte les erreurs qui ont pu se produire en cours de transmission et les récupère en demandant la retransmission des données.
- Elle réassemble les blocs de données dans l'ordre correct à l'extrémité destination, et les transmet à la couche session. **[Loa, Vèq, Znt 95]**

2- Type de connexion

Il existe deux types de connexions :

-Orienté connexion (fiable).
-Non orienté connexion (non fiable).

Le type de service est déterminé à la connexion.

La couche transport indique quel message appartient à quelle connexion, s'il y a plusieurs programmes qui s'exécutent simultanément dans beaucoup de machines (ce qui implique de multiple connexion pour chaque machine).

3- Détection et correction d'erreur de bout en bout

Vérifie que la donnée a été reçue, comme elle avait était émise; si ce n'est pas le cas, elle corrigera les erreurs ou formulera une requête pour que le paquet défectueux soit réèmis.

4- Segmentation et réassemblage de bout en bout

Assure le bon réassemblage des paquets traités par la couche réseau.

5- Contrôle de flux de bout en bout

Vérifie que le flot d'informations est régulié, cela nécessite un mécanisme de régulation de flux pour qu'une machine hôte rapide ne surcharge pas une machine lente.

6- Les protocoles utilisés pour la couche transport

Les protocoles suivants sont actuellement utilisés pour cette couche :

- TP0, 1, 2, 3 ou 4.
- TCP, UDP.
- Transmission Control Protocol (TCP).
- Sequenced Packet Exchange (SPX).
- Netware Core Protocol (NCP) chez Novell.

Les quatre premières couches sont les couches réseau. Elles transportent physiquement les données d'une application vers une autre, sans erreurs. Les trois autres, sont chargées de formater les informations, et de procurer des voies d'accès multiples à la même application.

X- La couche Session - Session Layer (C5)

Elle permet à des utilisateurs sur différentes machines d'établir des sessions entre eux (des connexions). Une session peut être utilisée pour la connexion à distance d'un terminal à un ordinateur, pour un transfert des données ou pour toute autre application.

La couche session est la première couche de l'architecture de réseau à ne pas être concernée par la communication des informations proprement dite. Elle a pour but d'établir des voies de communications logiques vers les applications qui traitent les données.

Un des services de la couche session concerne la gestion du dialogue. Les sessions peuvent autoriser le mode bi- ou unidirectionnel du trafic. Pour cela la gestion du tour de rôle ou la gestion du jeton est essentielle pour que les deux côtés ne lancent pas la même opération en même temps. Pour gérer ces activités, la couche session fournit des jetons qui peuvent être échangés. Seul le côté possédant le jeton peut effectuer l'opération critique.

Un autre service de session est la synchronisation, elle consiste à placer les entités de session dans un état connu des deux interlocuteurs en cas d'erreur. Cela peut se réaliser par l'insertion de points de test (points de reprise).

L'ouverture d'une session nécessite la négociation de plusieurs paramètres entre les utilisateurs des extrémités ; comme les règles de communications, choisir les protocoles à utiliser.

Cette couche offre la possibilité d'organiser les échanges en unités indépendantes. Elle offre aussi une structure de contrôle pour la communication entre applications. Elle établit, maintient et clôt les sessions entre les applications.
Notion d'activité: on peut la démarrer, l'arrêter, l'interrompre et la recommencer.
Quelques protocoles définis par la couche session : X225, T62, RPC.

1- Correspondance entre une session et une connexion de transport

Une session a pour but le transport des données ; comme la couche transport, mais elle offre également des services avancés utiles à certaines applications.

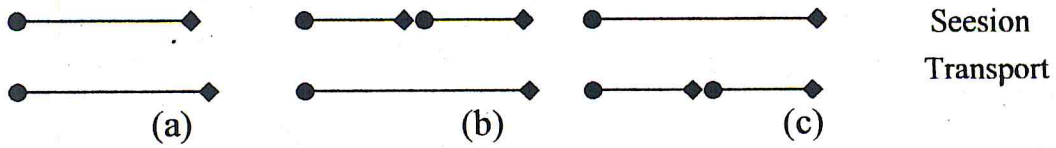


Figure 6 : La correspondance entre une session et une connexion de transport

La session et la connexion de transport sont similaires, bien qu'ils ne soient pas identiques. Trois cas peuvent se présenter :

a)- Il y a correspondance exacte entre une session et une connexion de transport.
b)- Plusieurs sessions successives sont établies sur une seule et même connexion de transport. Par exemple, ceci peut être utilisé dans une agence de voyage, dans laquelle chaque employé utilise un terminal relié à un ordinateur local ; celui-ci est relié à une base de données centrale de la compagnie pour enregistrer les réservations. Chaque fois qu'un employé veut faire une réservation, une session est ouverte avec l'ordinateur central, et elle est fermée lorsque la réservation est terminée. Mais il est inutile de libérer la connexion de transport sous-jacente, car celle-ci sera utilisée quelques instants plus tard par un autre employé.
c)- Plusieurs connexions de transport successives sont nécessaires pour une seule et même session. Ceci peut arriver lorsqu'une connexion de transport tombe en panne, la couche session établit alors une nouvelle connexion de transport de manière à poursuivre la connexion commencée.

XI- La couche Présentation - Présentation Layer (C6)

Elle permet de fournir aux utilisateurs ; les moyens d'exécuter les primitives de service de session, et une représentation des données, autrement dit une représentation qui ne dépend pas des ordinateurs, systèmes d'exploitation, etc. Elle fournit les moyens de spécifier des structures de données complexes, et inclut des services tels que, le cryptage, la compression et le formatage des données.
La couche présentation a pour but la mise en forme et la gestion des données, pour qu'elles soient compréhensibles par l'application de destination. Pour cela la couche présentation agit sur:
- La syntaxe.
- Les conversions de code et de jeux de caractères.
- La transformation des données, par exemple la compression (pour réduire le nombre de bits transmis).

Quelques protocoles définis par la couche présentation : x226, x208, ASNI.

En effet, il existe de multiples manières de coder les informations en informatique suivant le matériel et les logiciels utilisés. Par exemple:

- Plusieurs codes existent pour coder les caractères (ASCII, EBCIDC, etc).
- Les nombres peuvent être codés sur un nombre d'octets différents.
- Les octets de poids fort et de poids faible peuvent être répartis différemment, autrement dit, un nombre peut être lu de gauche à droite ou de droite à gauche.

XII- la couche Application - Application Layer (C7)

La couche application constitue le niveau le plus haut dans le modèle OSI. C'est l'interface entre l'utilisateur et le système OSI. C'est la où se situe vraiment les programmes (les applications) utilisant le réseau comme HTTP, FTP, Telnet et bien d'autres.
Cette couche joue un double rôle :

- Afficher les informations reçues.
- Envoyer les données fournies par l'utilisateur vers les couches inférieures.

Cette couche ne dispose pas de point d'accès SAP. Les processus d'application font partie des environnements système des utilisateurs et la couche application leur fournit les services nécessaires pour accéder à l'environnement OSI.
Les processus d'applications échangent leurs informations par l'intermédiaire des entités.
Exemple :
Terminal de réseau virtuel, transfère de fichier, courrier électronique, consultation des annuaires.

1- L'architecture de la couche

La couche application se compose de plusieurs éléments distincts appelés entités d'application, chacune d'elle est reliée à la couche présentation.
Chaque entité d'application contient des groupes de fonctions supportant une application particulière, ces groupes de fonctions sont appelés éléments de service d'application ASE.
Les éléments de service d'application sont des ensembles identifiables de fonction fournissant les services OSI.
Il y a deux types d'éléments de service d'application :

1- Commun (CASE) //est commun à plusieurs application.
2- Spécifique (SASE) //est spécifique à une application.

XIII- Conclusion

Nous avons vu dans ce chapitre l'interconnexion des systèmes ouverts, c'est le modèle de référence OSI, dont nous avons décrit son concept d'architecture en couche, ainsi que sa structure. Nous avons détaillé chaque couche par ordre croissant, en présentant la couche physique, la couche liaison de données, la couche réseau, puis la couche transport suivie de la couche session et présentation, et enfin la couche application.

Pour enchaîner, il faut parler du modèle OSI en tant que référence et introduire la suite de protocole TCP/IP.

Chapitre II :
Le protocole TCP/ IP (Transmission Control Protocol / Internet Protocol).

I- Introduction

TCP/IP(**T**ransmission **C**ontrol **P**rotocol/**I**nternet **P**rotocol) est un ensemble de protocoles organisé en couches. Il offre une méthode pour transférer des informations d'une machine à une autre. TCP/IP est un protocole de communication qui traite les erreurs survenant lors de la transmission, gère le routage et la livraison des données et contrôle la transmission elle même. Afin d'en comprendre la signification prenons l'exemple de l'envoi d'un message: Il y a tout d'abord un protocole de messagerie définissant l'ensemble des commandes qu'une machine envoie à l'autre, ces commandes indiquent qui est l'émetteur, à qui le message est destiné et enfin le texte du message. Ce protocole suppose donc l'existence d'un lien fiable entre les deux ordinateurs, c'est TCP qui assurera la fiabilité de la communication. Le protocole de messagerie est conçu pour utiliser les services de TCP qui effectuent la retransmission en cas d'erreurs, qui divise le message en plusieurs datagrammes et s'assure qu'ils arrivent correctement. Comme ces fonctions sont nécessaires à de nombreuses applications, elles constituent un protocole séparé ne faisant pas partie des spécifications de l'envoi de messages. On peut considérer TCP comme une bibliothèque de procédures mises à la disposition des applications lorsqu'elles ont besoin d'une communication fiable vers un autre ordinateur. De la même façon TCP fait appel aux services de IP, mais des applications peuvent faire directement appel à IP sans passer par TCP. En général les applications TCP/IP utilisent quatre couches:

1-couche application.
2-couche transport.
3-couche IP.
4-couche d'accès au réseau.

Figure 1 : Architecture de TCP/IP

La section suivante présentera la correspondance de TCP/IP avec le modèle OSI.
Dans la section 3, nous retrouverons le principe du fonctionnement de TCP/IP, enfin dans la section 4 nous détaillerons le fonctionnement des différentes couches de TCP/IP.

II- La correspondance avec le modèle OSI

- La couche application correspond aux 3 couches hautes du modèle de référence OSI les couches application, présentation et session.
- La couche transport TCP/IP est semblable à la couche transport OSI, son rôle est de fournire un service fiable de bout en bout en mode orienté connexion.
- La couche IP est semblable au service réseau sans connexion OSI son rôle est de déplacer les pâques de données distinctes sur un nombre quelconque de réseau.
- La couche d'accès au réseau correspond aux couches physiques et de liaison de donnée OSI.

[Tan 91]

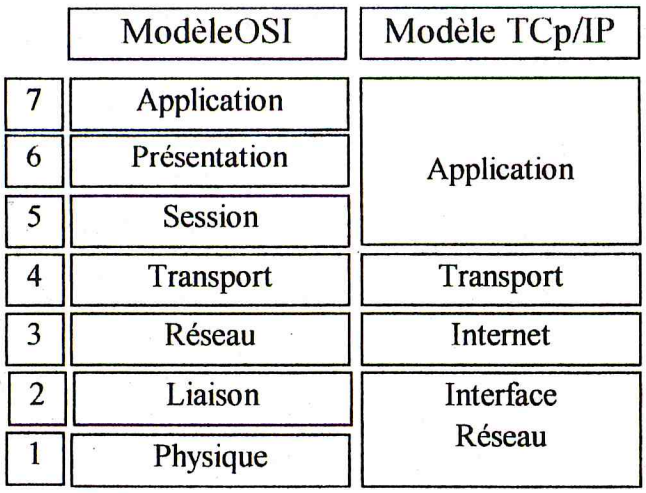


Figure 2 : Correspondance entre le modèle OSI et TCP/IP

III- Principe de fonctionnement

Lors d'une transmission, les données traversent chacune des couches au niveau de la machine émettrice. A chaque couche, une information est ajoutée au paquet de données, il s'agit d'un en-tête,(ensemble d'informations qui garantit la transmission).
Au niveau de la machine réceptrice, lors du passage dans chaque couche, l'en-tête est lu, puis supprimé. Ainsi, à la réception, le message est dans son état original.
A chaque niveau, le paquet de données change d'aspect, car on lui ajoute un en-tête, ainsi les appellations changent suivant les couches:

- Le paquet de données est appelé **message** au niveau de la couche application.
- Le message est ensuite encapsulé sous forme de paquet.
- Le segment une fois encapsulé dans la couche Internet (couche IP) prend le nom de **datagramme.**
- Au niveau de la couche accès réseau on parle de **trame.**

1- Caractéristiques de TCP/IP

L'architecture TCP/IP s'appuie sur des caractéristiques intéressantes parmi elle :

- TCP/IP est un protocole ouvert, il est transportable sur n'importe quel type de plate-forme.
- Le mode d'adressage est commun à tous les utilisateurs de TCP/IP quelle que soit la plate-forme qui l'utilise.
- Les protocoles de hauts niveaux sont standardisés ce qui permet des développements largement répandus sur tous types de machines. **[Puj 00]**

IV- Fonctionnement des couches du protocole TCP/IP

1- La couche accès au réseau

La couche accès réseau est la première couche de l'architecture TCP/IP, elle offre les capacités d'accéder à un réseau physique quel qu'il soit, c'est-à-dire fournir l'interface avec la technologie du réseau. Elle assure les mêmes services que la couche physique du modèle OSI.

2- Couche Internet

La couche Internet est responsable, de la transparence vis-à-vis de la topologie des réseaux connectés, que des supports physiques de transmission utilisés dans chacun des réseaux physiques qui constituent l'interconnexion. Pour ce faire, la couche Internet doit assurer les points suivants :

- Un niveau de service commun indépendant des supports physique interconnectés.
- Un mécanisme d'adressage global.
- Un modèle de routage pour transférer les données à travers l'interconnexion de réseaux physiques.
- Un mécanisme de fragmentation et de reassemblage des paquets échangés (la taille maximale d'un datagramme est 65535).

Toutes ces fonctionnalités sont fondamentales pour mettre en œuvre le concept d'interconnexion de réseaux. Ceci permet aux équipements de réseaux de voir une interconnexion comme étant homogène de nature.
La couche Internet comporte Plusieurs protocoles : ARP, IGMP, RIP, OSPF, IP (Internet Protocol) et ICMP (Internet Control Messages Protocol) en coopération pour réaliser la fonction de cette couche.

A . Le protocole IP

Le protocole IP fonctionne dans le mode sans connexion et fournit un service datagramme.
Il est caractérisé par les points suivants :

- C'est un protocole non fiable, il ne gère par la retransmission des datagrammes en cas d'erreurs.
- Il n'est pas connecté, il ne se soucie pas de savoir par quels nœuds passe un datagramme, ni même quelle sont ses machines de départ et d'arrivée.
- Il n'assure pas le contrôle de flux.

C'est un protocole de commutation de paquets qui s'occupe de l'adressage et du routage et de la fragmentation.

1- Principe de fonctionnement

TCP envoie chacun de ces datagrammes à IP, en donnant, bien sûr, l'adresse Internet de l'autre ordinateur (c'est tout ce que IP à besoin).
Son travail consiste simplement à trouver une route pour acheminer le datagramme à sa destination, il ajoute son propre en-tête pour permettre, aux passerelles (routeurs) ou autres systèmes intermédiaires, d'acheminer (d'aiguiller) le datagramme.
Les éléments principaux de l'en-tête IP pour l'acheminement des données sont:

- **L'adresse Internet source :** c'est l'adresse de la machine émettrice, et **l'adresse Internet destination :** c'est l'adresse de la machine réceptrice (elle permet, aux passerelles, de savoir la destination du datagramme), se sont des adresses sur 32 bits comme 128.6.4.194.
- **Le numéro de protocole :** il permet à l'autre IP de savoir que le datagramme est destiné à TCP ou UDP.
- **Le checksum (somme de contrôle) :** permet à l'IP récepteur de vérifier l'intégrité de l'en-tête.

2-L'adressage IP

Une adresse IP est codée sur 32 bits et se décompose en deux parties :
Un identificateur de réseaux qui référence un réseau physique particulier d'une interconnexion.
Un identificateur de la machine qui référence un équipement particulier de ce réseau. Une adresse IP identifie précisément l'endroit où un équipement est relié à une interconnexion. Ainsi, un équipement relié à plusieurs réseaux physiques aura plusieurs adresses (généralement une par accès). Un tel équipement est appelé machine « multidomicilier ».

Une adresse IP fait partie d'une classe. Cette dernière est définie en fonction du nombre de machines que nous souhaitons intégrer dans notre réseau.
Il existe 5 classes d'adresses possibles. La classe d'une adresse IP est définie en fonction de son premier octet.
Les classes A, B et C couramment utilisées que ce soit sur Internet ou dans le cadre des réseaux locaux ou privés.
La classe D est utilisée pour le multicast.
La classe E est réservée et non utilisée à l'heure actuelle.

La classe A : Cette couche est caractérisée par le premier bit qui est toujours à 0. Elle dispose de 7 bits pour identifier le réseau et de 24 bits pour identifier l'hôte. On a donc des réseaux de 1 à 127 et 2^{24} hôtes possibles.
Masque par défaut : 255.0.0.0

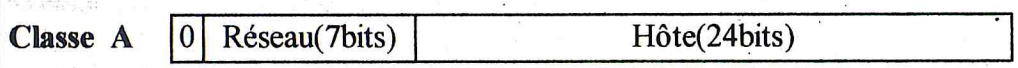


Figure 3 : La classe d'adressage A

La classe B : La classe B est caractérisée par les deux premiers bits qui sont à 1 0 et une adresse réseau sur 14 bits, et 16 bits pour identifier la machine. Ce qui fait 2^{14}= 16 384 réseaux (128.0 à 191.255) et 65 534 machines.
Leur premier octet est compris entre 128 et 191 inclus.
Masque par défaut : 255.255.0.0

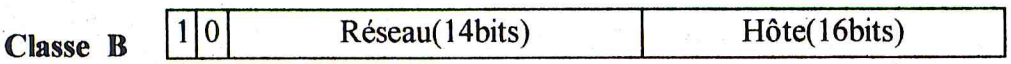


Figure 4 : La classe d'adressage B

La classe C : Est caractérisée par les trois premiers bits qui sont à 1 1 0 et une adresse réseau sur 21 bits, et 8 bits pour identifier la machine. Ce qui fait 2^{21} réseaux (de 192.0.0 à 223.255.255) et 254 machines.
Leur premier octet est compris entre 192 et 223.
Masque par défaut : 255.255.255.0

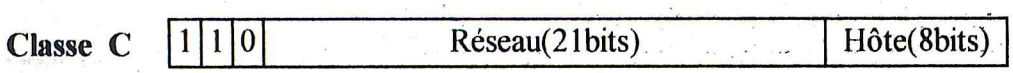


Figure 5 : La classe d'adressage C

La classe D : La classe D est caractérisée par les quatre premiers bits de l'adresse qui sont à 1 1 1 0, il s'agit d'une classe d'adressage spéciale. Cette classe est prévue pour faire du ``multicast", ou multipoint, ces adresses sont des adresses multidestinataires signifiant

que les datagrammes sont distribués vers un groupe d'ordinateurs, contrairement aux trois premières classes qui sont dédiées à l'``unicast" ou point à point.
Etendue d'une classe D : de 224 à 239.

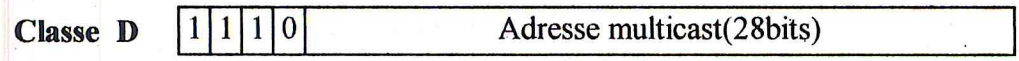


Figure 6 : La classe d'adressage D

La classe E : La classe E est caractérisée par les quatre premiers bits de l'adresse qui sont à 1111. Elle est réservée pour un usage futur.
Etendue d'une classe E : de 239 à 255.
Masque par défaut : 255.255.240.0 [Puj 00]

3- Spécification de IP

La figure suivante représente le paquet IP (datagramme) :
<---------- 32 bits ---------->

Version	Longueur d'en-tête	type de service	Longueur totale
Identification		Bourrage	Drapeau / Décalage fragment
Durée de vie		Protocole	Somme de contrôle en-tête
Adresse IP source			
Adresse IP destination			
Données			

Figure 7 : La spécification de IP

Bourrage : ce champ complète un datagramme qui n'est pas complet.
Version: identifie la version du protocole IP que l'on utilise (actuellement on utilise IPv4). Elle est codée sur 4 bits.
Longueur d'en-tête: il s'agit du nombre de mots de 32 bits sur lesquels est réparti l'en-tête.
Type de service: indique la qualité du service souhaitée pour le datagramme (priorité, délais, débit et fiabilité).
Longueur totale: indique la taille totale du datagramme en octets(entête et données). La taille de ce champ étant de 2 octets, la taille totale du datagramme ne peut dépasser 65536 octets.
Identification, drapeaux (flags) et déplacement de fragment sont des champs qui permettent la fragmentation des datagrammes.
Durée de vie: (appelée aussi TTL: Time To Live) indique le nombre maximal de routeurs à travers lesquels le datagramme peut passer. Ainsi ce champ est décrémenté à chaque passage dans un routeur, lorsque celui-ci atteint la valeur critique de 0, le routeur détruit le datagramme. Cela évite l'encombrement du réseau par les datagrammes perdus.
Protocole: identifie le protocole de haut niveau qui utilise IP.
Somme de contrôle de l'en-tête (header checksum): ce champ contient une valeur codée sur 16 bits qui permet de contrôler l'intégrité de l'en-tête afin de déterminer si celui-ci n'a pas été altéré pendant la transmission.
Adresse IP Source: adresse IP de la machine émettrice.
Adresse IP destination: Adresse IP du destinataire du message.

4-Fragmentation

Pour optimiser le débit, il est préférable que la totalité du datagramme IP soit encapsulé dans une seule trame de niveau 2 (Ethernet par exemple). Mais, comme un datagramme IP peut transiter à travers Internet sur un ensemble de réseaux aux technologies différentes, il est impossible de définir, à priori (lors de la définition du RFC), une taille maximale des datagrammes IP *MTU* (+) qui permette de les encapsuler dans une seule trame quel que soit le réseau (1500 octets pour Ethernet et 4470 pour FDDI par exemple).
Alors la fragmentation du datagramme devient nécessaire par le module Internet (passerelle, routeur…..).
Les procédures de fragmentation doivent pouvoir éclater un datagramme Internet en un nombre de "fragments" arbitraire et quelconque, selon la taille maximale MTU du réseau destinataire. La fragmentation se situe au niveau d'un routeur qui reçoit des datagrammes issus d'un réseau à grand MTU et qui doit les réexpédier vers un réseau à plus petit MTU.

Remarque :

1. La taille d'un fragment est choisie la plus grande possible tout en étant un multiple de 8 octets.
2. Un datagramme fragmenté n'est réassemblé que lorsqu'il arrive à destination finale, même s'ils traversent des réseaux avec un plus grand MTU, les routeurs ne réassemblent pas les petits fragments.
3. Chaque fragment est routé de manière totalement indépendante des autres fragments du datagramme d'où il provient.

Le processus de fragmentation-réassemblage :
Pour tenir compte de la fragmentation, chaque datagramme possède plusieurs champs permettant leur réassemblage:

Le champ "**Fragment Offset**" indique au récepteur la position du fragment reçu dans le datagramme original.
Le champ "**identification**" permet de reconnaître tous les fragments du même message (tous les fragments possèdent la même identification que le datagramme originale).
Le champ **longueur** indique la longueur total du fragment (cette longueur est recalculé pour chaque fragment).
Le champ **drapeau**: est composé de trois bits dont les deux derniers contrôlent la fragmentation:

- **DF**: *Don't Fragment* indique si le datagramme peut être fragmenté ou non.
- **MF**: *More Fragments*, en français fragments à suivre est mis systématiquement à 1 pour tous les fragments qui composent un datagramme sauf le dernier qui est mis à 0.

Remarques :

1. Le contenu des autres champs reste identique au datagramme original (identification, adresse source et destination, et le champ protocole)
2. Si un fragment doit être à nouveau fragmenté lorsqu'il arrive sur un réseau avec encore un plus petit MTU, ceci est fait comme décrit précédemment sauf que le calcul du champ déplacement de fragment est fait en tenant compte du déplacement inscrit dans le fragment à traiter.

A la réception, le destinataire final qui reçoit un premier fragment d'un datagramme, arme un temporisateur de réassemblage, c'est-à-dire un délai maximal d'attente de tous les fragments. Si ce délai est passé, et qu'il y a des fragments qui ne sont pas arrivés, alors il détruit les fragments reçus, et ne traite pas le datagramme. Plus précisément, l'ordinateur destinataire

décrémente à intervalles réguliers d'une unité, le champ TTL de chaque fragment en attente de réassemblage.

L'inconvénient du système de fragmentation :

L'inconvénient majeur de fragmentation est que si l'un des fragments ne parvient pas à destination, tout le paquet IP doit être retransmis. C'est pour cette raison qu'on évite d'utiliser la fragmentation en positionnant à 1 le flag don't fragment dans l'en-tête IP. Ainsi, si un routeur reçoit le paquet IP, et si la taille du paquet est supérieure à son MTU, il rejette le paquet et envoie à l'émetteur un message ICMP pour l'en informer.

Pour réassembler les fragments d'un datagramme Internet, un module Internet (par exemple dans un hôte destinataire) recombine les datagrammes dont les valeurs des quatre champs suivants sont identiques : identification, source, destination, et protocole. La recombinaison est réalisée en remplaçant la portion de donnée contenue dans chaque fragment, dans un tampon à la position relative indiquée par le champ "Fragment Offset" lu dans l'en-tête correspondant. Le premier fragment sera donc placé en début de tampon, et le dernier fragment récupéré aura le bit "Dernier Fragment" à zéro.

5- Le routage IP

Le routage concerne la recherche du meilleur chemin pour acheminer les paquets d'un point à un autre d'un réseau. Il permet de trouver de nouvelles routes en cas de panne, il se situe au niveau 3 du modèle OSI (la couche IP du modèle TCP), cette fonction est réalisée par des machines appelées routeurs.

Le routeur est un équipement relié à au moins deux réseaux. Il s'agit de machines ayant plusieurs cartes réseau dont chacune est reliée à un réseau différent. Ainsi un routeur n'a qu'à « regarder » sur quel réseau se trouve un ordinateur pour remettre les datagrammes venus d'une de ses interfaces en provenance de l'expéditeur.

D'une manière générale on distingue :

La **remise directe**, qui correspond au transfert d'un datagramme entre deux ordinateurs du même réseau (il suffit de comparer la partie "identificateur réseau" de l'adresse de destination à celle des adresse IP de la machine supportant l'entité IP locale).

La **remise indirecte :** quant au moins un routeur sépare l'expéditeur initial et le destinataire final. Il nécessite, de déterminer vers quel routeur envoyer le datagramme IP en fonction de sa destination finale et de la table de routage du routeur (le routeur consulte sa table de routage, une table qui définit le chemin à emprunter pour une adresse donnée).

Sur Internet le schéma est plus compliqué car:

- Le nombre de réseau auquel un routeur est connecté est généralement important.
- Les réseaux auquel le routeur est relié peuvent être reliés à d'autres réseaux que le routeur ne connaît pas directement.

Ainsi, les routeurs fonctionnent grâce à des tables de routage et des protocoles de routage, selon le modèle suivant:

- Le routeur reçoit des datagrammes provenants d'une machine connectée à un des réseaux auquel il est rattaché
- Les datagrammes sont transmis à la couche Internet.
- Le routeur regarde l'en-tête du datagramme pour déterminer si la remise est directe ou indirecte.
- Le routeur envoie le datagramme grâce à la carte réseau reliée au réseau sur lequel le routeur décide d'envoyer le paquet.

Types de routage :

1- Adaptabilité aux changements dans le réseau :

- **Routage statique**: (valable pour de petits réseaux) si la table de routage est entrée manuellement par l'administrateur, le chemin calculé est fixe.
  Un routeur statique doit connaître toutes les routes pour tous les réseaux et sous-réseaux auxquels il a une interface connectée.
  Les routeurs statiques ne se communiquent pas les nouvelles routes, ils n'échangent pas les informations de routage avec les routeurs dynamiques.
- **Routage dynamique**: la grande particularité des routeurs dynamiques c'est ; qu'ils échangent automatiquement entre eux les routes, vers des réseaux et sous-réseaux auxquels ils sont connectés.
  Dans ce cas le chemin peut varier en fonction de l'état du réseau: panne, nouvelle architecture du trafic sur le réseau, encombrements, etc…

2- Distribution du calcul sur le réseau:

- **Routage isolé**: chaque routeur calcule sa table.
- **Routage centralisé**: une seule machine centrale fait les calculs et transmet les résultats aux routeurs.
- **Routage distribué**: chaque routeur fait une partie du calcul et communique ses résultats aux routeurs voisins.

En pratique nous utilisons des algorithmes de routage distribués et dynamiques.

Remarque :

Pour les routeurs statiques et les routeurs dynamiques, tous les postes doivent connaître l'adresse de leur passerelle par défaut, qui correspond à l'IP de l'interface du routeur.

La table de routage :

Les algorithmes de routage IP utilisent sur chaque machine, une table de routage Internet (Internet Routing Table) aussi appelée table de routage IP (IP Routing Table).
La table de routage contient les informations relatives aux différentes destinations possibles et la façon de les atteindre.
Cette table est une table de correspondance entre l'adresse de la machine visée et le nœud suivant auquel le routeur doit délivrer le message. Il n'est pas nécessaire de stocker l'adresse IP complète de la machine, seul l'identificateur du réseau de l'adresse IP est stocké.

La technique qui consiste à ne connaître que l'adresse du prochain maillon menant à la destination, est appelée **routage par sauts successifs** (next-hop routing).

Parfois, il se peut que le destinataire appartient à un réseau non référencé dans la table de routage. Dans ce cas, le routeur utilise un **routeur par défaut** (passerelle par défaut).

Les protocoles de routage :

Sur Internet on trouve un ensemble de réseaux connectés, par conséquent tous les routeurs ne font pas le même travail, et fonctionnent donc avec des protocoles différents selon le type de réseau sur lequel ils se trouvent il y a :

- Les **routeurs noyaux** sont les routeurs principaux car ce sont eux qui relient les différents réseaux.
- Les **routeurs externes** permettent une liaison des réseaux autonomes entre eux. Ils fonctionnent avec un protocole appelé EGP (Exterior Gateway Protocol).
- Les **routeurs internes** permettent le routage des informations à l'intérieur d'un réseau autonome. Ils s'échangent des informations grâce à des protocoles appelés IGP (Interior Gateway Protocol), tels que RIP et OSPF.

Les protocoles de routage sont :
1- Basés sur l'algorithme de Vecteur de distance (ou Bellman-Ford, Ford-Fulkerson):
- Hello [RFC 891]: inutilisé.
- RIP (Routing information Protocol) [RFC 1058]: converge lentement vers un état stable, génère des boucles en cas de pannes de liaisons !
- IGRP
- EIGRP (Cisco).

2- Basés sur algorithme d'état des liaisons :
- OSPF (Open Shortest Past First) [RFC 1583]: performant: équilibrage de charge, sous-zones, routage par type de service, authentification des routeurs, complexe à mettre en œuvre.
- EIGRP (Cisco) (Il est Hybride) .

B.Le protocole ICMP (Internet Control Messag Protocol)

ICMP est un système de rapport d'erreurs. C'est une partie intégrante d'IP et il doit être inclus dans toutes ses implémentations. Il permet de générer des messages et des signaux d'erreurs, cohérentes et compatibles à destination des différentes versions d'IP et de plusieurs systèmes d'exploitation.
ICMP est le système de communication de la couche IP. Les messages générés par ICMP sont interprétés différemment par le logiciel de la couche IP.
Les messages ICMP ont un format semblable à celui de tout datagramme IP, et les datagrammes ICMP ne se distinguent en rien des datagrammes contenant des données, jusqu'à ce que la couche IP de la machine réceptrice appropriée les traite.

C. Le protocole ARP(Adress Resolution Protocol)

Étant donné que le protocole IP, et ses adresses, peuvent être utilisés sur des architectures matérielles différentes (réseau Ethernet, Token-Ring, ...) possédant leurs propres adresses physiques, il y a nécessité d'établir les correspondances entre adresses IP et adresses matérielles des ordinateurs d'un réseau lors du passage des datagrammes de la couche IP vers la couche interface réseau. Ceci est l'objectif des protocoles ARP (*Adress Resolution Protocol*).
ARP fournit une correspondance dynamique entre une adresse IP connue et l'adresse lui correspondant, RARP faisant l'inverse.
ARP fournir à une machine donnée l'adresse physique(matérielle) d'une autre machine située sur le même réseau à partir de l'adresse IP de la machine destinatrice.

La technique d'ARP :
Le module ARP de la machine émettrice envoie une requête ARP dans une trame Ethernet ou autre, selon le protocole utilisé dans la couche physique avec une adresse de destination multicast. Ainsi, toutes les machines du réseau local reçoivent cette requête contenant l'adresse IP à résoudre.
La couche ARP de la machine visée reconnaît que cette requête lui est destinée et répond par une réponse ARP contenant son adresse matérielle. Les autres machines du réseau ignorent la requête.
La réponse ARP est reçue par l'émetteur de la requête. Pour ce retour, il n'y a pas de problème de résolution puisque l'adresse physique de l'émetteur, étant envoyée dans la requête, elle est connue de la machine qui répond.
Pour éviter la multiplication des requêtes ARP, chaque machine gère un cache dans lequel elle mémorise les correspondances adresses IP / adresses physique déjà résolues préalablement.

Ainsi, le module ARP ne lancera une requête que lorsqu'il ne trouvera pas cette correspondance dans le cache. Mais, les correspondances ne sont pas conservées indéfiniment car, cela pourrait provoquer des erreurs lorsque l'on change un ordinateur (ou une carte réseau) sur le réseau, en conservant un même numéro IP pour cet ordinateur, mais évidemment pas la même adresse physique.
Chaque ligne du cache correspond à un composant et contient 4 informations le concernant.

- Index IF : le numéro de l'interface physique.
- Adresse physique : c'est l'adresse physique du composant.
- Adresse IP : l'adresse IP correspondante à l'adresse physique.
- Type : c'est le type d'entrée dans le cache ARP.

3- La couche transport

La couche transport TCP/IP correspond à la couche transport du modèle de référence OSI.
Les protocoles des couches précédentes, permettaient d'envoyer des informations d'une machine à une autre. La couche transport permet à des applications (tâches) tournants sur des machines distantes de communiquer (établir une connexion de bout en bout « c'est à dire de processus à processus » entre deux utilisateurs ou hôtes de l'inter-réseau).
Pour identifier ces applications, la couche transport utilise les numéros de **ports.**
La couche transport TCP/IP doit aussi pouvoir transférer des données de façon fiable à travers un ou plusieurs sous-réseaux dont la qualité est variable.
La couche transport comporte essentiellement deux protocoles de transport :

A. **TCP :** un protocole **orienté connexion** qui assure le contrôle des erreurs.
B. **UDP :** un protocole **non orienté connexion** dont le contrôle d'erreur est archaïque.

La majorité des services réseau fonctionnent avec le protocole TCP (**T**ransmission **C**ontrol **P**rotocol). Lors d'une communication à travers le protocole TCP, les deux machines doivent établir une connexion, la machine émettrice est appelée le client, la machine réceptrice est appelée serveur. On dit alors que l'on est dans un environnement client-serveur. TCP fournit un système relativement fiable appelé **acquittement de bonne réception avec retransmission.** Pour éviter la perte éventuelle d'information entre les hôtes. **[Loa, Vèq, Znt 95]**

3-1- Les protocoles de la couche transport

- **Le protocole TCP (orienté connexion)**

Le protocole de contrôle de transmission TCP est le protocole en mode connexion de la famille des protocoles Internet. Comme TCP est un protocole orienté connexion, il comporte trois phases distinctes :

- L'établissement de la connexion.
- Le transfert des données.
- La libération de la connexion.

Le principe de fonctionnement (le niveau TCP) :
Au départ on a un seul flot de données que l'on essaye d'envoyer de longueur arbitraire.
Cette couche accepte ces données et les découpes en blocs de donnée plus petite, selon la taille négociée de chaque extrémité TCP (choisissant la plus petite si chaque réseau extrême supporte une taille différente).
TCP ajoute un en-tête d'au moins 20 caractères au début de chaque segment. Les éléments les plus importants de cet en-tête sont :

- Le numéro de **port source**.

- Le numéro de **port destination**.
- Le **numéro d'ordre**, il est utilisé par l'autre extrémité pour rassembler les segments dans le bon ordre, et qu'il n'en manque pas.

La figure suivante représente le schéma d'un segment TCP.

<table>
<tr><td colspan="2">0</td><td colspan="2">31</td></tr>
<tr><td colspan="2">Port source</td><td colspan="2">Port destination</td></tr>
<tr><td colspan="4">Numéro d'ordre</td></tr>
<tr><td colspan="4">Numéro d'acquittement</td></tr>
<tr><td>Position des données</td><td colspan="2">Reservé
U/A/P/R/S/F
R/C/S/S/Y/I
G/K/T/N/N</td><td>Fenêtre</td></tr>
<tr><td colspan="3">Checksum</td><td>Pointeurs données urgentes</td></tr>
<tr><td colspan="4">Données</td></tr>
</table>

Figure8 : Segments TCP

Pour s'assurer que le segment est bien arrivé, il faut que le récepteur renvoie un **acquittement** sous la forme d'un segment contenant un numéro d'acquittement. Par exemple, si ce numéro est 500 cela signifie que le récepteur a tout reçu jusqu'au 500 octets. L'émetteur renvoie les données s'il ne reçoit pas d'acquittement dans un délai raisonnable.
Note:il n'y a pas d'accusé négatif (NACK), c'est une temporisation(Time-out) qui joue ce rôle.
La fenêtre est utilisée pour contrôler la taille des données qui peut être envoyée à chaque instant. Elle indique la quantité de données qu'elle peut absorber. Lorsqu'une machine est en réception, l'espace restant dans sa fenêtre diminue, lorsqu'il devient nul, l'émetteur doit s'arrêter, lorsque les données reçues sont traitées, le récepteur augmente la taille de sa fenêtre pour indiquer qu'il est en mesure d'en recevoir d'autres. On utilise souvent le même segment pour acquitter et également autoriser l'envoi de nouvelles données (en augmentant la taille de la fenêtre).
Données urgentes : permet à une extrémité de forcer l'autre à traiter un octet particulier, cela permet de gérer les évènements asynchrones comme l'envoi d'un caractère de contrôle ou d'une commande qui interrompe les sorties.
Le checksum (somme de contrôle), c'est un nombre calculé en ajoutant tous les octets du paquet(segment), le résultat est mis dans l'en-tête, TCP de l'autre extrémité fait le même calcul, s'ils ne sont pas égaux le segment est rejeté.

Pour assurer le suivi de toutes les connexions, TCP utilise une table de connexion. Chacune des connexions existantes possède une entrée dans la table, contenant des informations sur la connexion. La figure ci-dessous montre l'agencement de la table de connexion.

	Etat	Adresse locale	Port local	Adresse distante	Port distant
Connexion1	établit	193.194.81.42	1050	193.194.81.33	23
Connexion2					
Connexion3					
Connexion4					

Tableau 1 : La table de connexion TCP

La signification de chaque colonne est :

- Etat : c'est l'état de la connexion.
- Adresse locale : c'est l'adresse IP de la connexion.

- Port local : c'est le numéro de port de la connexion locale.
- Adresse distante : c'est l'adresse IP de la machine distante.
- Port distant : c'est le numéro de port de la connexion distante.

- **Le protocole UDP (Le mode non connecté)**

Le protocole UDP (**U**ser **D**atagram **P**rotocol) utilise IP pour acheminer les données d'un ordinateur à un autre, en mode non connecté(non fiable).
UDP n'utilise pas d'accusé de réception et ne peut donc pas garantir que les données ont bien été reçues. Il ne réordonne pas les messages si ceux-ci n'arrivent pas dans l'ordre dans lequel ils ont été émis et il n'assure pas de contrôle de flux.
C'est donc à l'application qui utilise UDP de gérer les problèmes de perte de messages, duplications, retards, déséquencement,... (parce que le récepteur n'est pas apte à faire face aux flux de datagramme qui lui arrivent).
UDP fournit un service supplémentaire par rapport à IP, il permet de distinguer plusieurs applications destinatrices sur la même machine par l'intermédiaire des ports.

Port UDP sousrce	Port UDP destination
Longueur	Checksum
Données	

Figure9: Structure d'un datagramme

Le format détaillé d'un datagramme UDP :

- **Les numéros de ports** (chacun sur 16 bits) identifient les processus émetteur et récepteur.
- **Le champ longueur** contient sur 2 octets la taille de l'en-tête et des données transmises. Puisqu'un datagramme UDP peut ne transmettre aucune donnée, la valeur minimale de la longueur est 8.
- **Le checksum** est un total de contrôle qui est optionnel, car il n'est pas indispensable lorsque UDP est utilisé sur un réseau très fiable. S'il est fixé à 0 c'est qu'en fait, il n'a pas été calculé.

3-2- Adressage des applications

Pour effectuer l'adressage des applications, TCP et UDP gèrent des valeurs entières, non signées, codées sur 16-bits, appelées **'Ports'**. Les numéros de ports inférieurs à 512 sont affectés par l'IANA, ce sont les ports réservés. Les numéros de ports supérieurs à 1024 sont disponibles aux utilisateurs.
La notion d'adressage des applications constitue un exemple du multiplexage des protocoles :

- Au niveau de la couche interface, chaque réseau physique distingue habituellement ces clients(entités au niveau de la couche Internet), grâce aux différentes valeurs du champ **type** de l'entête Ethernet (ainsi, **Ethernet** utilise la valeur 0x0800 pour designer **IP**).
- Au niveau de la couche Internet, IP distingue les clients à l'aide de différentes valeurs du champ **protocole** de l'entête IP (ainsi, **IP** utilise la valeur 17 pour designer le protocole **UDP**).
- Au niveau de la couche transport, TCP et UDP distinguent les clients (les entités de la couche application) au moyen des **numéros de ports** différents (ex : **UDP** utilise la valeur décimale 161 et 162 pour designer **SNMP).**

La combinaison d'une adresse IP et d'un numéro de port est appelée une 'prise' (**socket**), elle identifie d'une façon unique une entité d'application s'exécutant dans une interconnexion de réseaux.

3-3-Les Sockets

Les Sockets (prises de raccordement) offrent aux programmeurs une interface entre le programme d'application et les protocoles de communication. Les Sockets forment un mécanisme de communication bidirectionnel inter-processus dans un environnement distribué. Elle est définit par une adresse réseau (IP) et un numéro de port (associé à une application sur une machine), utilisé pour envoyer ou recevoir des données. C'est donc l'API (Application Programing Interface) TCP/IP qui assure le passage des données de la couche application vers les couches inférieures.

Une application distribuée fonctionne selon le modèle **client-serveur**. Pour réaliser le déroulement d'une communication entre le client et le serveur à travers le réseau, les processus client et serveur utilisent les services de la couche transport, par une interface bien définie. Dans l'environnement Windows, cette interface est définie par **Winsock** API (Windows Sockets Application Programming Interface).
L'interface Winsock définit un ensemble d'opérations, qui peuvent être appelés par le processus client ou serveur, par exemple ; pour établir la communication avec un processus pair ou ; pour transmettre ou recevoir des données, chaque opération est réalisée par une fonction ou un groupe de fonctions.

Remarque : Nous allons revenir sur le détail des sockets dans l'annexe B.

4- La couche application

La couche application est la couche située au sommet des couches de la pile TCP/IP.
La couche application fournit les protocoles et les fonctions nécessaires pour les applications utilisateur. Ces applications communiquent grâce à un des deux protocoles de la couche inférieure (la couche transport) TCP ou UDP.

Parmi les applications de cette couche :
-SNMP : protocole simple de gestion de réseau.
-FTP : protocole de transfert de fichier.
-SMTP : protocole simple de transfert de Courier.
-TELNET : réseau de télécommunications, aussi appelée NVT 'Terminal virtuel de réseau'.
On peut aussi concevoir d'autres applications répondants à des besoins utilisateurs spécifiques. Les applications spécifiques utilisateurs sont souvent partagées par des entreprises de même secteur d'activité.

4-1- Protocole SNMP

SNMP (**S**imple **N**etwork **M**anagement **P**rotocol) sert, à échanger des informations de gestion du réseau, surveiller des processus et des périphériques, utiliser la forme sans connexion du protocole de transporte ; appelée protocole datagramme utilisateur UDP.

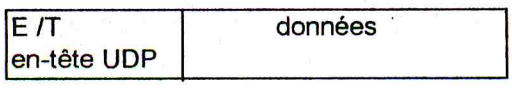


Figure 10 : Le message SNMP utilise la forme sans connexion

Remarque : Nous allons bien détailler ce protocole ultérieurement.

4-2- Protocole FTP

Est un protocole de transfert de fichiers " **F**ile **T**ransfert **P**rotocol " il est utilisé pour le transfert du fichier à distance.

4-3- Protocole SMTP

SMTP (**P**rotocole **S**imple de **T**ransfert de **C**ourier) : Il est utilisé pour le transfert du courrier électronique.

4-4- Protocole TELNET

TELNET est un protocole de terminal distant, il permet la connexion à distance.

V- Conclusion

Nous avons abordé dans ce chapitre le protocole TCP/IP et sa correspondance avec le modèle OSI, nous avons parlé également de son principe de fonctionnement, puis détaillé chaque couche de son architecture séparément.
Le domaine des réseaux est très vaste, un réseau ou un ensemble de réseaux interconnectés peut contenir différents types d'éléments.
Pour que l'ensemble fonctionne bien et soit fiable, il est nécessaire d'avoir des outils (fonctions et protocoles...) pour pouvoir gérer et administrer le réseau. C'est le thème de notre prochain chapitre.

Chapitre III : Protocole Simple de Gestion de Réseaux SNMP(Simple Network Management Protocol)

I- Introduction

Gérer, c'est vouloir tirer la meilleure partie de la structure que l'on gère. Cependant, ce système est dual car la conception d'une gestion dépend de la structure gérée. Inversement, le comportement futur de cette structure dépendra fortement de sa gestion.
La gestion des réseaux est un ensemble d'activités liées au contrôle, à la coordination et la surveillance des ressources qui participent à la communication.
Gérer un réseau revient à observer son activité (c'est-à-dire collection des statistiques sur le débit réel, calcul de taux d'erreurs...), et à contrôler les opérations en cours (comme le contrôle des accès et le contrôle de statut des connections etc.), et à agir sur l'ensemble de ses ressources de communication (exemple : activer, initialiser une station ou un routeur) grâce au gestionnaire.

1- Les attendus d'une administration de réseau

Les besoins fonctionnels sont regroupés autour de cinq grandes fonctionnalités :
Gestion des fautes : elle est constituée de trois grandes fonctionnalités :

- La fonction de surveillance qui consiste essentiellement à monitorer les pannes, à prendre en compte les événements non sollicités, à créer et à examiner les journaux, valider (confirmer) les fautes.
- La fonction de localisation des pannes qui consiste à déterminer le ou les éléments en cause, tenant compte des phénomènes d'effet de bord pouvant survenir. Cela est généralement fait par des recherches dichotomiques, et par l'utilisation d'outils de test.
- La fonction de détermination des pannes qui s'effectue par l'analyse de journaux, et par le choix d'actions curatives.

Gestion des configurations et des noms : trois grandes fonctionnalités la caractérisent :

- La fonction d'installation : consiste à paramétrer un composant, à l'initialiser et à le mettre en service ou à le supprimer. Cette fonction permet le contrôle et la mise à jour des différents paramètres, et tient à jour les différents paramètres et les différentes versions implémentées.
- La fonction de contrôle et de surveillance qui consiste à surveiller et à vérifier les changements d'état et certains aspects sur demande.
- La fonction de gestion des noms qui permet de localiser sans ambiguïté tout élément dans un réseau.

Gestion de performances : les fonctionnalités qui la caractérisent sont :

- La fonction de surveillance dont l'objectif est de collecter des données statistiques et des fautes.
- La fonction de gestion de trafic qui se base sur des données précédentes et qui permet de réguler le trafic écoulé dans le système.
- La fonction d'observation de la qualité de service qui consiste à évoluer les paramètres qui composent, par exemple, l'évaluation du délai d'un appel, du succès ou de l'échec d'une connexion et de la qualité de cette connexion.

Gestion de la comptabilité : elle a quatre grandes fonctionnalités :

- La charge des ressources dont le rôle est de déterminer et de surveiller la charge admissible des ressources.
- Le coût des ressources détermine le prix d'utilisation des ressources.
- La facturation récupère l'ensemble des informations de coût imputable à un utilisateur.

- La gestion des limites utilisateur qui consiste à déterminer et à surveiller les quotas d'utilisation des ressources par les utilisateurs.

Gestion de sécurité : ces fonctions sont :

- La gestion de confidentialité par l'utilisation des clés d'encryption et de mécanismes d'authentification.
- L'audit par la maintenance et l'examen des journaux, permettant de surveiller les tentatives de connexion intempestives.
- L'enregistrement et la gestion des abonnés pour maintenir et déterminer leurs droits de connexion.

Il existe différents protocoles permettant de transmettre les informations des équipements au gestionnaire. Cela permet d'identifier les éléments raccordés au réseau et de connaître leur état. Parmi ces protocoles le protocole SNMP.

2- Historique

Le protocole de gestion de réseaux, SNMP (**S**imple **N**etwork **M**anagement **P**rotocol, traduisez protocole simple de gestion de réseau) est dérivé du protocole SGMP (**S**imple **G**ateway **M**onitoring **P**rotocol). Il a été Créé en 1988 par l'IETF (**I**nternet **E**ngineering **T**ask **F**orce) dans le cadre de la définition d'un système de gestion des réseaux utilisant les protocoles TCP/IP. SNMP a été approuvé par l'IAB (Internet Activities Board), responsable des spécifications de TCP/IP. L'architecture de SNMP fut définie dans le RFC 1067 et ratifiée comme standard Internet dans le RFC 1098.
SNMP visait deux objectifs : l'une est de fédérer en un standard unique des protocoles multiples, liés chacun à un type d'équipement, ou à une marque pour centraliser et simplifier la gestion des réseaux, l'autre est d'assurer rapidement une large diffusion grâce à une mise en oeuvre simple.
Ce protocole fait aujourd'hui l'unanimité dans la communauté Unix, et son adoption par les grands de l'informatique comme Digital, HP et IBM renforce sa pérennité.

3- Introduction

SNMP est un protocole, comme son nom l'indique, pour effectuer de la gestion de réseau.
Il permet, de contrôler un réseau à distance en interrogeant les stations qui en font partie, sur leur état et, de modifier leurs configurations, faire des tests de sécurité et, observer différentes informations liées à l'émission de données.
Il peut même être utilisé pour gérer des logiciels et bases de données à distance. Depuis qu'il est devenu un standard TCP/IP, son utilisation a beaucoup augmenté. D'ailleurs, il est le protocole le plus utilisé pour gérer des équipements de réseau (routeurs, ponts, etc.) et beaucoup de logiciels de gestion de réseau sont basés sur ce protocole.
Ce protocole se situe dans la couche application des deux modèles OSI et TCP/IP.

Le protocole SNMP permet de répondre à un grand nombre de besoins, parmi ces besoins:
- disposer d'une cartographie du réseau.
- fournir un inventaire précis de chaque machine.
- mesurer la consommation d'une application.
- signaler les dysfonctionnements par des alertes.

Les avantages du protocole SNMP :
- protocole très simple et facile à utiliser.

- permet une gestion à distance des différentes machines.
- le modèle fonctionnel pour la surveillance et pour la gestion est extensible.
- indépendant de l'architecture des machines administrées.

La section suivante présentera l'architecture de SNMP, elle sera suivie par son principe de fonctionnement dans la section 3 et puis la section 4, introduira la spécification du message SNMP. La section 5, décrira le processus d'envoi du message SNMP, suivi par les versions de SNMP. Le format des messages SNMP sera présenté dans la section 7, suivi de la section 8, qui définira la communauté SNMP, ensuite dans la section 9, nous présenterons les commandes SNMP, puis les différentes opérations dans la section 10. Les tables MIBs seront définies dans la section 11 de ce chapitre.
La section 12, portera sur l'agent SNMP intelligent, et la deuxième version de SNMP sera décrite dans la section 13. La section 14, présentera les nouveautés de SNMPv3, suivi par son architecture dans la section 15. Le traitement et le transport des messages seront décrit dans la section 16 et enfin, la section 17, présentera la sécurité dans SNMPv3.

II- Architecture de SNMP

L'approche SNMP a retenu le modèle gestionnaire/agent qui se compose de :
- **Eléments de réseaux:** appelés aussi nœuds administrés MN (Managed Node). Ce sont les équipements à gérer sur le réseau (stations de travail, routeurs, concentrateurs, ponts, etc..).
- **Agents :** Les agents SNMP sont placés dans les équipements compatibles SNMP. Se sont des programmes qui s'exécutent dans les éléments du réseau, ils fournissent des informations à partir de la MIB, dont les éléments reflètent à tout moment l'état de l'objet qu'ils représentent.

Deux agents particuliers existent :

- L'agent Rmon qui est situé dans une sonde, se comporte comme un analyseur de protocole local, et peut déclencher des alertes de performances.
- Le Proxy-agent qui sert d'intermédiaire entre un ou plusieurs autres agents et le manager. Il est utilisé soit pour une conversion de protocole (SNMP <=> protocole propriétaire), soit pour regrouper les informations provenant d'un réseau local distant. Il ne fait remonter vers le manager que les conditions anormales et permet ainsi de limiter le trafic du réseau d'interconnexion, souvent à bas débit.

- **Stations d'administration :**(NMS: Network Management Station) au moins une. C'est la station qui exécute un programme de gestion SNMP. Son but principal est de contrôler les stations du réseau et de les interroger sur différentes informations. Elle se doit d'avoir un processeur rapide avec beaucoup de mémoire et d'espace disque (pour archiver les informations).
- **Un protocole réseau** (Le protocole SNMP) utilisé par la NMS et les agents pour échanger des informations d'administration. Cela signifie que les requêtes envoyées aux éléments de réseau, ainsi que les réponses associées retournées à la NMS auront le format du protocole SNMP.

Remarque : Le nœud (MN) sur lequel s'exécute l'agent est appelé **client**. Les agents possèdent des paramètres et des compteurs qui sont appelés **objets** ou **attributs**.

Les systèmes SNMP utilisent deux éléments clés pour la transmission des informations :

- La **SMI** (Structure of Management Information), définit comment sont représentées, dans la MIB, les informations relatives aux objets de gestion et comment elles sont obtenues.

- Les **MIB**s (Management Information Base) représentant la base de données de l'ensemble des objets pour un agent donné. Elle regroupe l'ensemble des variables relatives aux matériels et aux logiciels supportés par le réseau, et définit les objets de gestion dans l'environnement TCP/IP.

La figure suivante montre l'environnement SNMP:

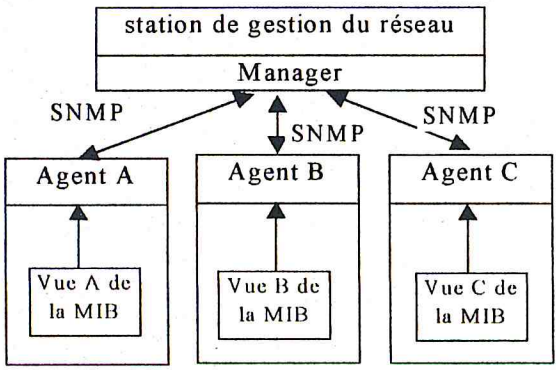


Figure 1 : L'environnement SNMP

L'élément de réseau accepte et traite toutes les demandes d'accès aux informations qui sont stockées dans sa base d'information MIB qui sont émises par la NMS.

SNMP utilise le protocole UDP (**U**ser **D**atagramm **P**rotocol) pour transporter les données entre les agents et le manager. Il travaille donc en mode non connecté, ce qui peut poser des problèmes de sécurité des échanges (aucune garantie de livraison).
SNMP a l'avantage d'être simple, cependant il a des capacités très limitées au niveau sécurité, principalement pour l'authentification. Tous les systèmes SNMP doivent également supporter le protocole IP pour transporter les données entre les agents et les stations de gestion.

III- Principe de fonctionnement de SNMP

SNMP fonctionne avec des requêtes, des réponses et des alertes. Pour observer le fonctionnement des agents, les stations de gestion **NMS** envoient des requêtes à l'agent sur chaque élément du réseau et celui-ci doit exécuter la requête (exécuter certaines taches) et envoyer sa réponse qui contient les informations requises depuis la **MIB**. Il peut aussi y avoir des alertes asynchrones venant des agents lorsqu'ils veulent avertir la NMS d'un problème tel que des erreurs de transmission. Mais ces alertes spontanées sont limitées.
Les managers effectuent une interrogation périodique des agents de manière à vérifier leurs états. La structure des paquets est définit en utilisant la syntaxe ASN.1 (Abstract Syntax Notation One).

IV- Spécifications du message SNMP

La taille maximale d'un message SNMP est de 484 octets, on lui attache l'entête UDP qui fait 8 octets ensuite l'entête IP qui fait 20 octets. Sa longueur totale devient 512 octets.

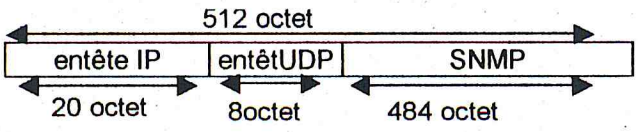


Figure 2: Spécification du message SNMP

V- Le processus d'envoie d'un message SNMP

SNMP réside généralement sur le port bien connu numéro 161, sur ce port SNMP écoute les requêtes provenant de la NMS.
La NMS envoie des datagrammes à travers l'inter-réseau au processus SNMP de l'élément de réseau situé sur le port numéro 161.

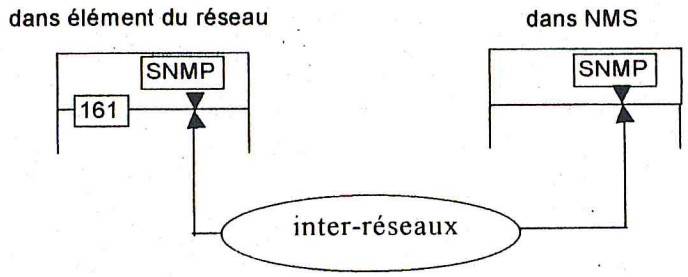


Figure 3 : Le processus d'envoie d'un message SNMP

La NMS maintient ouvert un port distinct afin de pouvoir détecter d'éventuelles alertes c'est le port numéro 162.

VI- Les versions de SNMP

Il existe plusieurs versions du protocole SNMP :
SNMPv1 (complet) : c'est la version la plus utilisée.
SNMPsec : cette version a ajoutée de la sécurité au protocole SNMPv1 (elle est vite oublier).
SNMPv2 : la version 2 est beaucoup plus complexe que la version 1; elle contient entre autres un niveau hiérarchique d'administration, ce qui permet d'avoir un administrateur central et de petits NMS dans le réseau. Elle contient une gamme de message d'erreur plus vaste, utilise un nouveau type de tables MIB, les MIBs II, qui contiennent plus d'éléments. Elle engendre aussi un niveau plus élevé de sécurité. Cette version n'a pas remplacé la première version à cause de ça complexité. On trouve dans cette version :

- **SNMPv2p** : ses travaux ne portaient pas seulement sur la sécurité mais aussi sur une mise à jour des opérations du protocole, et des nouveaux types de données…
- **SNMPv2c** : c'est une amélioration des opérations de protocole et des types d'opérations du protocole SNMPv2p.
- **SNMPv2u** : elle utilise les opérations et les types de données de SNMPv2c.
- **SNMPv2*** : elle combine les meilleures parties de SNMPv2p et SNMPv2u.

SNMPv3 : son objective est de résoudre le problème de sécurité qui est relative (puisque aucun algorithme de sécurité n'est parfait et que la technologie de décodage évolue). Le standard SNMPv3 propose que les logiciels SNMP soient bâti de façon modulaire, c'est à dire que certains modules peuvent être ajoutés et enlevés facilement par l'utilisateur. Comme l'architecture SNMP est modulaire, le standard SNMPv3 est mis à jour sans entraîner des changements dans tous les documents du standard et sans trop de difficultés.
Cette version comprend " un module de sécurité plus élevée, un module de traitement des messages, des modules d'applications et un répartiteur des paquets ". Elle permet aussi d'interagir avec les anciennes versions.

VII- Le format des messages SNMP échangés entre la NMS et l'agent de gestion

Un message SNMP (i.e SNMPv1 quand on ne précise pas la version) possède la structure suivante : **[Lau 96]**

Version	Communauté	PDU

Figure 4 : Format des messages SNMP

La version : indique le numéro de version utilisé.
La communauté : la communauté est décrite par une chaîne de caractère, elle indique le non de la communauté du réseau (par défaut PUBLIC), qui aide à identifier les messages SNMP.

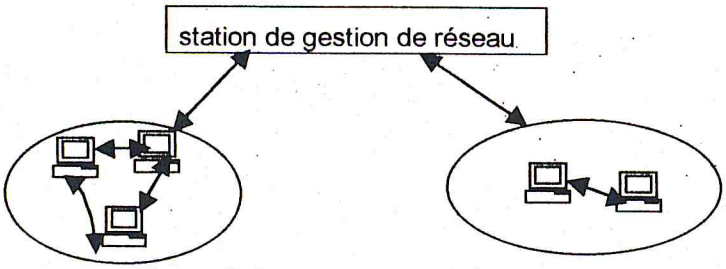


Figure 5: La communauté du réseau

PDU (Protocol Data Unit traduisez unité de données de protocole): il existe 5 types différents :
- La PDU GetRequest
- La PDU GetNextRequest
- La PDU SetRequest
- La PDU GetRespons
- La PDU Trap

1- Structure des PDU set et get

La structure des PDU set et get est la suivante :

Type de PDU	ID demande	Etat d'erreur	Index d'erreur	Var bind List

Figure 6 : Structure des PDUs get et set

1- **Type de PDU** :
0 :GetRequest
1 :GetNextRequest
2 :GetResponse
3 : SetRequest

2- **ID demande** : contient un entier, il fait correspondre une réponse à une demande particulière et aide à la détection d'erreurs, dans le cas où deux réponses arrivaient à la NMS dans le même laps de temps avec le même identifiant de requête.

3- **Etat d'erreur** : ce champ est uniquement utilisé par l'agent dans l'envoi de la PDU GetRespons à la NMS. Il indique le statut de la requête précédemment envoyée par la NMS à l'agent. Ce champ « état d'erreur » contient une valeur qui varie entre 0 et 5 :

Si « état d'erreur » vaut 0 en réponse à une PDU de demande, il n'y a aucune erreur : la demande à été exécuté avec succès.
Si l'état d'erreur vaut entre 1 et 4, alors pour une raison quelconque la demande n'a pas pu être exécutée avec succès.

Les types de réponses erronées de SNMP :

Réponse	Description
NoAccess	accès non permis
WrongLengh	(erreur de longueur)
WrongValue ()	valeur erronée
Wrongtype	type erroné
WrongEncoding	erreur d'encodage
NoCreation	objet non créé
ReadOnly	pas de permission d 'écriture
NoWritable	pas de permission d 'écriture
AutorisationError	erreur d'autorisation

Tableau 1 : Les types de réponses erronées de SNMP

Si l'état d'erreur vaut 5, une erreur quelconque, non spécifiée s'est produite, ce qui a empêché la bonne exécution de la demande.
4- Index d'erreur : ce champ est uniquement utilisé dans la PDU GetRespons afin de fournir plus d'informations sur l'erreur détectée par l'agent. Cet index pointe sur la première variable du champs VarBindList ayant causé l'erreur, sa valeur étant la position de cette variable dans la liste (le décompte commençant à partir de 1).
5- Var bind list : (variable binding liste c'est à dire liste de liaison de variable), c'est la liste de toutes les variables contenant les objets sur lesquels s'opère la requête. C'est une variable qui renferme le couple [ObjetID,Valeur] où « ObjetID » est l'identifiant de l'objet dans la MIB et «Valeur » sa valeur dans la MIB.

Remarque : Dans les **PDU(get request)** et **(get next request)** les champs qui voudront toujours zéro sont :

- état d'erreur
- index d'erreur

2- La structure de PDU d'interruption ou PDU trap

La PDU trap permet de signaler des évènements qui se produisent sur le réseau et qui sont causés par des erreurs.
La structure de la PDU (trap) est la suivante :

Type de PDU	Entreprise	Agent	Interrupt générique	Interrupt spécifique	Horodatage	VarBindList

Figure 7 : Structure de PDU Trap

1- **Type de PDU :** dans ce cas toujours égal à 4.
2- **Entreprise :** permet de spécifier le type d'objet qui a généré l'interruption, il contient l'identification de l'objet qui est dans l'arbre d'enregistrement hiérarchique.
3- **Adresse agent** : ce champ contient l'adresse IP de l'agent, ce qui permet à la NMS d'identifier la provenance du trap qu'elle reçoit.

4- Interruption générique : c'est un entier représentant l'une des valeurs de trap prédéfinies en standards pour le protocole SNMP. Ces valeurs ou traps génériques sont au nombre de 7 :

0.démarrage à froid.
1.démarrage à chaud.
2.liaison interrompue.
3.liaison rétablie.
4.échec d'identification.
5.perte de voisinage EGP.
6.spécifique entreprise.

5- Interruption spécifique : Si le champ interruption générique vaut 6 (spécifique entreprise), alors le message d'interruption est spécifique à cette communauté de gestion de réseau.

Exemple pour lequel l'élément de réseau et sa NMS pourraient s'accorder :

- Ratio erreur / trafic dépassé.
- Saturation de passerelle.
- Temps de réponse maximum de l'hôte dépassé .
- Nombre maximum de retransmission sur la liaison.

6- Champ horodatage : ce champ contient l'heure à laquelle le trap a été génèré. C'est aussi le temps qui s'est écoulé depuis la dernière initialisation de l'agent (cette valeur est exprimée en centième de seconde).

7- VarBindList : liaison avec variable, ce champ se compose d'une liste de variables et de valeurs (des informations) associées à l'interruption.

VIII- Communauté SNMP

La communauté SNMP est une relation entre un agent SNMP et une ou plusieurs stations d'administration SNMP, qui définit l'authentification, le contrôle d'accès. Une communauté SNMP est caractérisée par son nom qui est une simple chaîne de caractères, appelée le **nom de la communauté SNMP** qui est utilisé pour offrir un mécanisme très simple de sécurité entre les agents et la station d'administration.

Les messages SNMP échangés entre les agents et les stations d'administration SNMP sont composés de deux parties:

- le nom de la communauté SNMP et d'autres informations nécessaires pour valider que l'entité SNMP émise fait bien partie de la communauté spécifiée,
- les données, contenant une opération SNMP et les opérandes associées.

1- Authentification

L'authentification se fait de façon très simple. Le nom de la communauté est placé en clair dans le message SNMP, ce nom fonctionne comme un mot de passe. Si le nom de la communauté correspond à une communauté définie au niveau de l'objet, l'entité SNMP émise est considérée comme étant **authentifiée** comme un membre de la communauté.

2-Autorisation

Une fois que l'entité SNMP émise est authentifiée comme un membre de la communauté, le nœud sur lequel se trouve l'agent doit déterminer à quel niveau d'accès peut se placer la requête SNMP.

Pour chaque objet, la communauté définit un mode d'accès qui peut être soit « lecture seulement », soit « lecture/écriture ».

Remarque : la vue de la MIB et le mode d'accès forment pour chaque objet **le profil de la communauté** (comunity profile). Ainsi pour chaque objet, le profil de communauté définit les opérations qui sont autorisées sur cet objet.

IX- Les commandes SNMP

SNMP est un protocole de type question / réponse asynchrone. Par conséquent, une entité SNMP n'a pas besoin d'attendre une réponse après avoir envoyé un message.
Les commandes SNMP sont les suivantes :

- la PDU (get request: demande de lecture) permet de connaître la valeur d'une variable dans la MIB. Par exemple: nous pouvons envoyer une PDU (get request) à une passerelle afin de mesure le niveau du trafic sur un itinéraire particulier.
- la PDU (get next request) permet de connaître la valeur de la variable suivante dans une liste de la MIB; un objet peut posséder une liste de variables auxquelles la NMS veut accéder une par une. SNMP n'a pas à gérer les adresses et les emplacements de ces objets.
- la PDU (set request) permet de modifier la valeur d'une variable de la MIB. Par exemple: nous utilisons une PDU set request pour modifier la durée de vie des datagrammes envoyés par un hôte. Si la valeur est bien modifiée un message (get response) le confirmera.
- la PDU (get response) est une réponse aux PDUs get request et get next request, mais aussi set request, elle contient les données demandées par la NMS.
- la PDU (trap) permet de signaler à tout moment des erreurs qui se produisent sur le réseau. Par exemple: si un des systèmes du réseau tombe en panne, l'agent envoie un message à la NMS pour l'informer pour entreprendre telle ou telle action.

X- Les opérations

On distingue quatre types d'opérations au niveau SNMP:
Get-request / get-reponse : la station d'administration interroge l'agent.

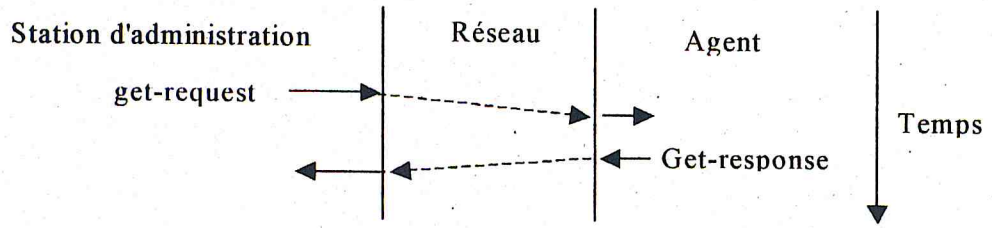


Figure 8 : Interaction (Administrateur/Agent), opération get

Get-next-reques / get-response : la station d'administration interroge une table de l'agent.

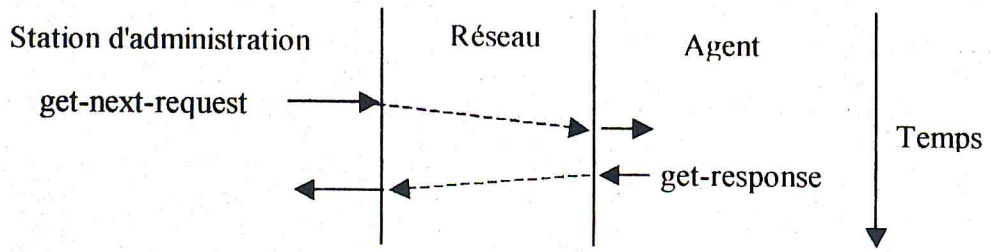


Figure 9 : Interaction (Administrateur/Agent), opération get-next

Set-request / get-response : la station d'administration enregistre des données dans un agent.

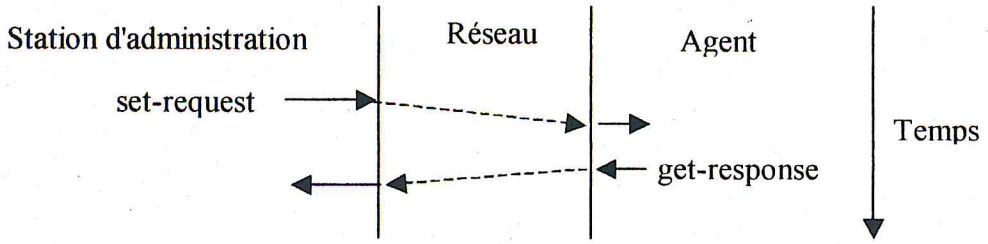


Figure 10 : Interaction (Administrateur/ Agent), opération set

Trap: l'agent envoie un événement "extraordinaire" (Alerte) vers la ou les stations d'administration.

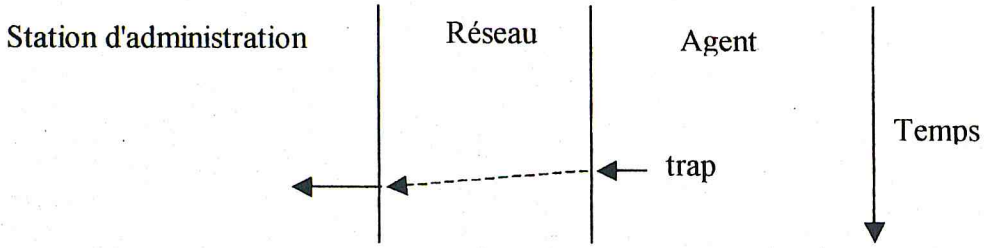


Figure 11 : Interaction (Administrateur/ Agent), opération trap

Remarque : Les fonctions sont acheminées par UDP par le port 161 pour les get/ set et le port 162 pour les traps. **[Loa 96]**

XI- Les Tables MIBs

1- Définition

La MIB est une base de donnée faisant référence à des objets clairement définis, mais les informations ne se trouvant pas stockées réellement dans une base de données gérée par SNMP. La MIB peut être considéré comme une base de données virtuelle ou une grille de repérage pour savoir de quel objet / variable l'on parle. La valeur des objets est stockée quelque part et c'est au programme serveur (daemon SNMP) de les retrouver lorsqu'ils sont réclamés par le gestionnaire. Une MIB est une collection de tous les objets que maintient un agent donné.
Pour qu'un client (nœud sur lequel se trouve l'agent) accède à ces objets, il faut qu'il soit au courant de leur existence. Une MIB contient un certain nombre d'informations standards : **c'est la MIB standard**. Or pour la plupart des éléments réseau, on rajoute un certain nombre d'objets propres à un agent pour en exploiter les possibilités : **c'est la MIB privée**.
Par exemple, dans la MIB standard il y a des compteurs qui gèrent les paquets émis ou reçus sur chaque interface de l'appareil. Dans ce cas n'importe quel client est capable de lire ces compteurs, des constructeurs différents sont capables de retrouver ces informations.
Une centaine de Mibs privées ont été crées. Pour cela, le vendeur enrichit les fonctionnalités de ses agents SNMP d'une collection d'objets propriétaire tout en démarquant son marketing produit. Aussi ces Mibs privées deviennent-elle vite indispensables pour affiner la gestion d'un équipement spécifique.
Par exemple, les sociétés suivantes ont des MIBs privées d'entreprise :

- **IBM** 1.3.6.1.4.1.2,

- **DEC** 1.3.6.1.4.1.36,
- **SUN** 1.3.6.1.4.1.42,
- **CHIPCOM** 1.3.6.1.4.1.49

Remarque : L'extension et l'enrichissement de la MIB est faite par l'IAB (Internet Activities Board), organisme gérant les RFC (Request For Comment) avec comme règle que tout nouvel objet doit être défini avec un nouveau nom, les noms déjà utilisés ne peuvent resservir à une nouvelle définition, ils peuvent devenir "obsolètes", c'est-à-dire inutilisés mais existant toujours dans la base.

2- Structure de la MIB

Le premier standard utilisé pour la définition des objets d'administration de la MIB standard fut la MIB-I. La structure est la suivante :

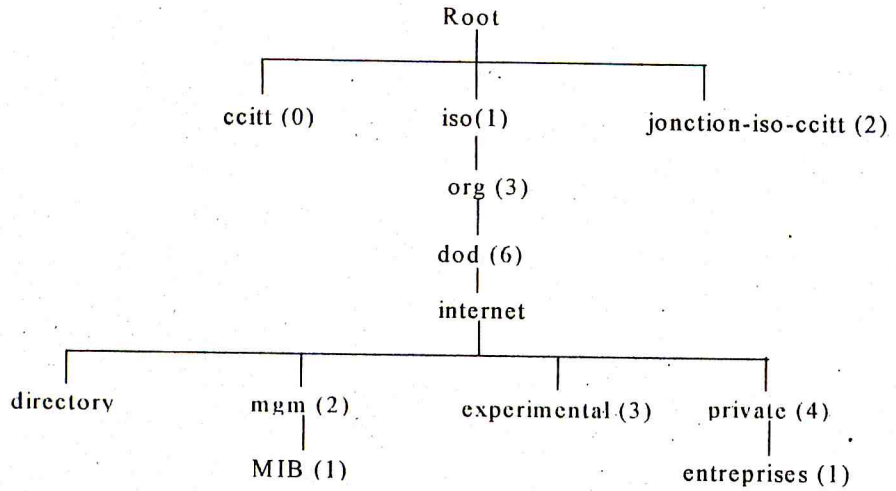


Figure 12 : Structure de la MIB

Signification de quelque objets:
Iso :Internationnl standard organisation
dod : Department Of Defence
directory :repertoire
private :privé

Le tableau suivant indique les composantes de la MIB I :

Numéro	Objet	Nombre de sous-objets
1	system	3
2	interfaces	23
3	At	3
4	Ip	33
5	icmp	26
6	Tcp	17
7	Udp	4
8	Egp	6

Tableau 2 : Les composants de la MIB I

Dans une MIB toutes les informations sont répertoriées en nombres entiers (représentant des noms) dans une architecture hiérarchisée en respectant la syntaxe ASN.1 (*Abstract Syntax*

Notation One). Donc on utilise la syntaxe ASN.1 pour décrire les données. Chaque objet est représenté par un "object identifier".
Par exemple :
- pour accéder à un objet d'administration, son identificateur (chemin qui conduit à l'objet) commencera par : iso-identified-org-dod-internet-mgt Ou 1.3.6.1.2
- 1.3.6.1.2.1.1.3.0 est l'élément, ou OID, *sysUpTime*, qui contient le temps écoulé depuis la mise en marche de l'appareil.

3- Les MIBs-2

Fin 1989, la MIB 2 apparaît par le RFC 1158. La MIB 2 est un sur-ensemble de la MIB 1. Elle comporte 171 objets ce qui fait qu'elle a 57 objets de plus que la MIB1, qui n'en avait que 114. Les deux MIBs peuvent cohabiter ensemble. Ces nouveaux objets permettront de décrire de façon plus fine les fonctionnalités d'un équipement. Par exemple, dans le sous groupe «système» on peut trouver le nom de la personne à contacter en cas de défaut sur un équipement.

La définition de la MIB 2 est la suivante :

Numéro	Objet	Nombre de sous-objets
1	system	7
2	interfaces	23
3	At	3
4	Ip	38
5	Icmp	26
6	Tcp	19
7	Udp	7
8	Egp	18
9	cmot	0
10	transmission	0
11	Snmp	30

Tableau 3 : Les composants de la MIB II

Ces objets réfèrent :

Nom des groupes	Description
System	Information générales concernant l'agent à travers le système
Interfaces	Informations concernant chaque interface IP de l'agent
At	la table de translation d'adresses qui réalise la correspondance entre l'adresse MAC et l'adresse IP
Ip	Compteurs IP
Icmp	Compteurs ICMP
Tcp	Compteurs TCP
Udp	Compteurs UDP
Egp	Compteurs EGP
Snmp	Statistiques du trafic SNMP
CMOT	Compteur pour CMOT (Common Management Over TCP/IP) protocole OSI équivalent à SNMP.

Tableau 4 : Description des composants des deus MIBs

4- La syntaxe ASN.1

4-1- Domaines d'utilisation

A l'origine ASN1 est normalisée pour spécifier les données des protocoles de l'interconnexion de systèmes ouverts (OSI), mais, depuis elle a été adoptée par de nombreux autres domaines:

- la gestion de réseaux (SNMP)
- la messagerie électronique X.400
- l'annuaire X.500
- la téléphonie cellulaire (MAP - Mobile Application Part)
- les réseaux intelligents
- le commerce et le paiement électroniques : protocoles SET (*Secured Electronic Transaction*), authentification X.509...
- les communications multimédias : norme MHEG, visioconférence (série de normes UIT-T T.120), communications multimédias en temps réel sur Internet (norme H.323)...
- le contrôle aérien (ATN - Aeronautical Telecommunication Network)
- elle est aussi la notation de typage de données d'autres langages formels comme GDMO (*Guidelines for the Definition of Managed Objects*) dans le domaine de la gestion des réseaux, LDS (*Langage de Description et de Spécification*) pour spécifier le comportement de systèmes télécommunicants et TTCN (*Tree and Tabular Combined Notation*) pour l'écriture de suites de tests de protocoles.

4-2- Définitions formelles en ASN.1

ASN.1 est un langage formel qui permet de représenter de manière universelle une information. Comme la représentation des données (entiers, flottants, chaînes de caractères...) est spécifique à chaque machine, il est nécessaire de disposer d'une représentation commune pour échanger ces données. C'est l'une des raisons du succès d'ASN.1 car cette notation est associée à plusieurs ensembles de règles de codages normalisés comme les BER (*Basic Encoding Rules*), ou plus récemment les PER (*Packed Encoding Rules*) pour les utilisations où la bande passante est un objectif prioritaire. Ces règles de codage permettent de sérialiser les données sous forme de chaînes d'octets à transmettre.
Toutes les variables MIB doivent être définies et référencées à l'aide de la notation ASN.1. Elle présente deux caractéristiques principales: une notation utilisée dans les documents manipulés par script et une représentation codée de la même information, utilisée dans les protocoles de communication. Dans les deux cas la notation formelle précise et élimine toutes les ambiguïtés.
Les protocoles d'administration de réseaux utilisent cette notation formelle pour définir le nom et le type des variables de la base de données d'informations d'administrations de réseaux
L'environnement de gestion utilise ASN1 pour deux choses :
- définir les structure des PDUs échangées par le protocole de gestion,
- définir les objets gérés.

Description de données :

-Déclarations ASN.1 similaires aux déclarations en C

- La déclaration de type dans ASN.1 : NouveauType ::= TypesExistants
- La déclaration d'une variable dans ASN.1 : nomDeLaVariable ::=valeur.

-Types de données: Chaque type possède un identifiant unique le TAG. La notation offre un certain nombre de types prédéfinis tels que :

- Les types de données simples
- Les types de données simplement construites

- Les types de données applicatives.

Les types de données simples :

- INTEGER : un nombre positif ou négatif.
- OCTET STRING : chaîne de caractères, chaque caractère peut prendre une valeur entre 0 et 255.
- OBJECT IDENTIFIER : c'est une suite de nombres entiers séparés par un point, il traduit la place d'un objet dans l'arbre d'information.
- NULL : un emplacement vide.

Les types de données simplement construites :

SEQUENCE : elle est utilisée pour constituer des listes, elle contient un certain nombre d'éléments dont chacun est un autre type ASN.1, elle peut être vide.

SQUENCE-OF : elle est utilisée pour constituer des tables, elle peut être vide.

Les types de données applicatives : elles sont introduites de manière implicite à partir des structures simples.

IpAdress : adresse IP sur 32 bits.

Counter32 : compteur incrément de 0 à $2^{32}-1$, remise à zéro lorsque le maximum est atteint.

Gauge32 : compteur spécifique qui peut croître ou décroître à la différence de Counter32. Il varie de 0 à $2^{32}-1$.

TimeTicks : compteur de temps dont l'unité de base est un intervalle exprimé en centième de sencondes, incrément jusqu'à $2^{32}-1$.

XII- Agent SNMP Intelligent

Agent intelligent est un agent qui a des tâches supplémentaires autres que de fournir des informations. L'un de ces agents est l'agent RMON (Ressource network MONitoring).

1- Agent RMON

Le Remote network MONitoring est un concept bâti pour servire des informations concernant les couches basses (comme le trafic d'un réseau) à partir d'un agent SNMP au manager. Il sert à visualiser le trafic réseau d'un sous réseau LAN dans son ensemble. Ce qu'un agent SNMP simple ne permet pas (analyser seulement le trafic d'un agent).

La mesure du trafic (l'analyse du trafic) sur un réseau s'effectue par des appareils spéciaux appelés analyseurs, sondes ou probes (en anglais), ils sont souvent cachés dans des concentrateurs ou Hubs. Les informations collectées par ces appareils peuvent être multiples, comme le nombre de paquets transitant sur le LAN par seconde, le nombre de collisions existant sur le réseau (pour Ethernet), les erreurs etc. De plus on peut analyser certains paquets particuliers sélectionnés selon des caractéristiques choisies ou les filtrer selon les besoins.

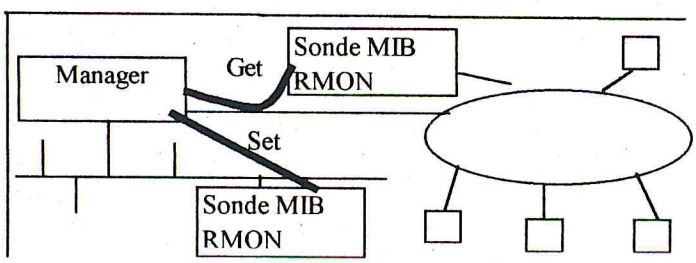


Figure 13 : L'agent RMON travail en liaison avec un manager

Un analyseur se focalise sur l'activité d'un sous-réseau et peut travailler d'une manière autonome ou en liaison avec un manager d'une façon 'remote'. Dans ce dernier cas le mécanisme de liaison peut se faire à travers SNMP, c'est à dire que la sonde est un agent SNMP intelligent appelé agent RMON, interrogeable par un manager SNMP.

L'agent RMON doit avoir une certaine intelligence ; dû au travail de collecte d'informations de la sonde remote, qui peut être 'coupée' de son manager, et dû à la volonté de réduire le trafic entre le manager et cette sonde. Donc elle va pouvoir travailler par elle-même afin de faire une collecte d'informations, celle ci lui a été spécifiées à travers des variables MIB; elle peut de plus réagir par l'émission de traps lorsque ces informations atteignent un seuil (paramétrables à travers les variables MIB).

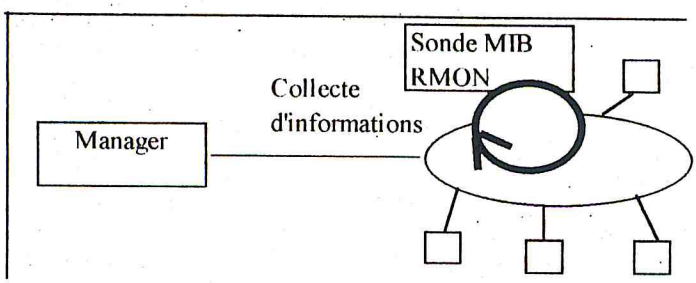


Figure 14 : L'agent RMON travail d'une manière autonome

Tout le paramétrage des fonctions à fournir et la collecte des informations se fait à travers des variables MIB RMON.

2- MIB RMON

La MIB RMON est une extension de la MIBII. On lui a attribuée le 'subtree identifier' 16, c'est-à-dire que la MIB RMON a comme OID 1.3.6.1.2.1.16.

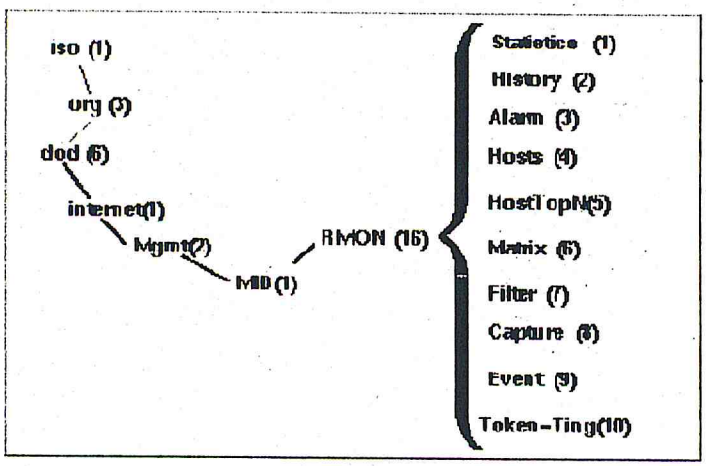


Figure 15 : Les identificateurs d'objets de la MIB RMON

Les différents groupes sont organisés suivant le principe des tables SNMP; il y a des tables de contrôles (Control Tables) pour paramétrer et définir la collecte d'informations et des tables de données (Data Table) pour récupérer les résultats informatifs (utiles).
La signification de ces différents groupes : **[RFC 1271]**

- **Statistics** "Statistique" c'est une simple table de données 'data', maintenant une information de bas niveau (physique) sur les statistiques d'erreurs et d'utilisation du sous-réseau que la sonde analyse.
- **History** "Historique" mémorisation périodique des extractions de statistiques de la table statistique.
- **Alarm** Positionnement d'intervalles et de seuils déclencheur pour générer des traps en fonction des informations statistiques.
- **Hosts** contient les compteurs des trafics venant ou allant vers les hosts du segment.
- **HostTopN** contient les statistiques hosts triées sur critères sélectés venant du groupe précédent. C'est une liste des N entités les plus actives dans un délai programmable.
- **Matrix** présente l'utilisation sous forme matricielle, afin que l'on puisse combiner deux à deux ces informations.
- **Filter** permet d'observer les paquets suivant un filtre, ce qui éventuellement permet de les capturer lorsque cette information est arrivée au 'channel'. Ces filtres jouent sur les 'data' (bit pattern) ou sur les status (valid, error . . .).
- **Capture** précède à l'envoi des paquets filtrés vers la station de Management.
- **Event** traite tous les événements générés par l'agent RMON, en particulier les alarmes qui peuvent donner lieu à des traps.
- **Token Ring** est l'adaptation des tables précédentes dans le contexte Token Ring.

Ces différents groupes ne sont pas tous obligatoires dans un agent RMON, mais il existe un principe de dépendance comme l'Alarm group a besoin de l'Event group, le HostTopN group n'existe que si le Host Group est là, le Capture group est lié à la présence du Filter group.

Cette mise en place de MIB dans un agent permet d'obtenir facilement des informations du réseau.
Exemples :
-Pour pouvoir détecter rapidement quel est le plus gros consommateur du réseau, nous utilisons le groupe (HostTopN).
- Nous utilisons le groupe(Matrix) pour savoir quelles sont les liaisons privilégiées entre telle ou telle machine.
-Le groupe (Alarm, Event) est utilisé pour avertir l'utilisateur des erreurs, dans ce cas l'utilisateur peut les détecter par le groupe (Filter) et les analyser (Capture). Et peut savoir aussi, comment se fait-il que ces erreurs de tel type soient aussi fréquentes par le groupe (Statistics).
Pour exploiter cette MIB RMON, des applicatifs sont nécessaires sur la station gestionnaire pour paramétrer et afficher ces sondes RMON intelligentes. En règle générale l'affichage se traduit par une vision graphique des flux d'informations ou de la réception des traps émis par la sonde.
RMON est donc un outil intelligent qui aide à résoudre les problèmes liés à la gestion de réseaux.

3- RMON 2

Malgré l'utilité et l'intérêt de RMON, il a été constaté des faiblesses à cette technique. Celles-ci sont liées à son implémentation. RMON ne s'adresse qu'aux deux premières couches du modèle OSI (Physique et Data Link), ce qui a pour conséquence qu'une RMON 1 ne peut analyser que le segment où il se trouve, et cette analyse se fait au niveau MAC (couche 2 Data Link). La reconnaissance des protocoles et de son adressage ne peut se faire dans RMON que si on lui adjoint des groupes de MIB qui s'adressent aux couches supérieures du modèle OSI.

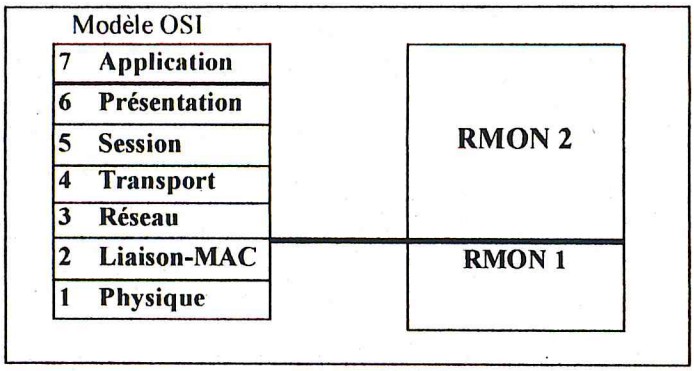


Figure 16 : RMON1 et RMON2 dans le modèle OSI

Pour cela neuf nouveaux groupes ont été créés pour prendre en compte le dépassement de la couche MAC.
Protocol Directory définit les protocoles que la sonde peut analyser.
Protocol Distribution prend les statistiques suivant le protocole.
Address Mapping fait la relation entre l'adresse MAC et l'adresse du protocole (exemple: MAC->IP).
Network layer Host mesure globale des trames suivant le protocole.
Network layer Matrix mesure entre deux hosts (pas forcément dans le même segment).
Application layer Host monte vers les couches hautes de l'OSI pour faire l'analyse.
Application layer Matrix mesure entre deux hosts (suivant le protocole applicatif).
History mémorise les statistiques de niveau 3 en local.
Probe Configuration normalisation de la configuration d'une sonde à partir du Manager.

Conclusion :
Le principe important à retenir de cet Agent RMON évolutif est que l'on donne de l'intelligence à un agent SNMP réputé instrumental à la base, c'est-à-dire qu'originellement il ne devait répondre qu'aux demandes ou au positionnement des variables MIB émises par le manager; mais, désormais, cet agent peut agir éventuellement sans l'aide de son manager et faire une collecte d'informations et réagir sur cette collecte d'une manière autonome. Cette solution a donné naissance au principe de proxy-agent ou sous-agent SNMP qui travail dans une station de travail sous l'agent SNMP.

XIII- SNMPv2

1- Historique

Au début des années 1992, l'Internet Engineering Task Force (IETF) réclame des propositions d'améliorations de SNMPv1. Dans le même temps une équipe formée par Jeffrey Case, Keith McCloghie, Marshall Rose et Steven Waldbusser préparent un nouveau protocole nommé the Simple Management Protocol (SMP). La fusion de ces deux recherches sur SNMP donne SNMPv2, dont les caractéristiques sont expliquées dans les RFC allant du numéro 1441 à 1452 (mi 1993).

2- Faiblesses de SNMPv1

Bien que SNMPv1 est simple à mettre en œuvre, peut chère, stable, performant (puisqu'il est rapide et de petite taille) et il a une implémentation rapide, il présente tout de même des faiblesses qui se résument en quatre problèmes :

1.Protocole gourmand :
Le fonctionnement de base de SNMP est le «pooling» pour obtenir des informations du réseau, ce qui fait de SNMP un protocole «bavard», grand consommateur de bande passante. Si un élément recherché se trouve en vingtième position dans une table MIB, cette procédure implique quarante échanges. Pour diminuer la charge, il a été rajouté à SNMPV2 de nouvelles opérations et de nouveaux critères d'erreurs dans les messages afin d'éviter une surcharge du réseau, dû à la communication entre agent et gestionnaire.
2.Protocole unique (pour TCP / IP seulement) :
La gestion de réseau s'adresse à des organisations qui utilisent plus d'un protocole, donc SNMP est trop restreint dans son mode de fonctionnement d'un protocole unique TCP / IP. C'est pourquoi les RFC prenant en compte d'autres modes de transport que IP tel que RFC1418 (OSI), RFC1419 (AppleTalk Datagramm Delivery Protocol DDP) et RFC1420 (Novell's Internetwork Packet eXchange) ont été intégré dans la logique de SNMPv2.
3.Sécurité :
La sécurité au moment de la description de SNMP n'avait pas été prise en considération, car la gestion des éléments de fonctionnement d'un réseau ne semblait pas être un élément crucial de sécurité. Donc SNMPv1 a été bâti sur une logique de mot de passe (community name) inséré dans chaque message (mais en toute clarté) par conséquent la falsification est très facile. SNMPv2 remédie à cette logique et crée un concept de partie pouvant être utilisé à d'autres fins qu'uniquement sécuritaire.
4.Gestionnaire à Gestionnaire :
Le protocole SNMPv1 doit son succès à sa simplicité, et celle-ci vient en partie de la communication, qui s'établit entre un Gestionnaire et son Agent avec un petit nombre d'opérations. SNMPv2 dépasse cette simplicité et montre sa possibilité d'établir des liaisons de Gestionnaire à Gestionnaire, afin de montrer qu'il peut gérer des réseaux importants, ne se réduisant pas à un seul et unique gestionnaire.

3- Les nouveautés de SNMPv2

3- 1- Le nouvel arbre "Internet OID"

SNMPv2 a rajouté un certain nombre de branches à l'arbre de « l'internet OID » (Object Identifier) pour définir les différents objets dont ce protocole va avoir besoin.
Cette augmentation dans l'arbre consiste en une branche spécifiquement sécurité :1.3.6.1.5 et en une autre SNMPv2 :1.3.6.1.6. Sous cette dernière, il se trouve trois sous-groupes qui s'occupent du mode de transport (Transport Domain) **snmpDomain**, de la gestion des proxys **snmpProxys** et des modules de la MIB de SNMPv2 **snmpModules**. Cette dernière branche a des sous-groupes pour la MIB SNMPv2, la gestion de Manager à Manager (M2M) et celle des parties (partyMib).

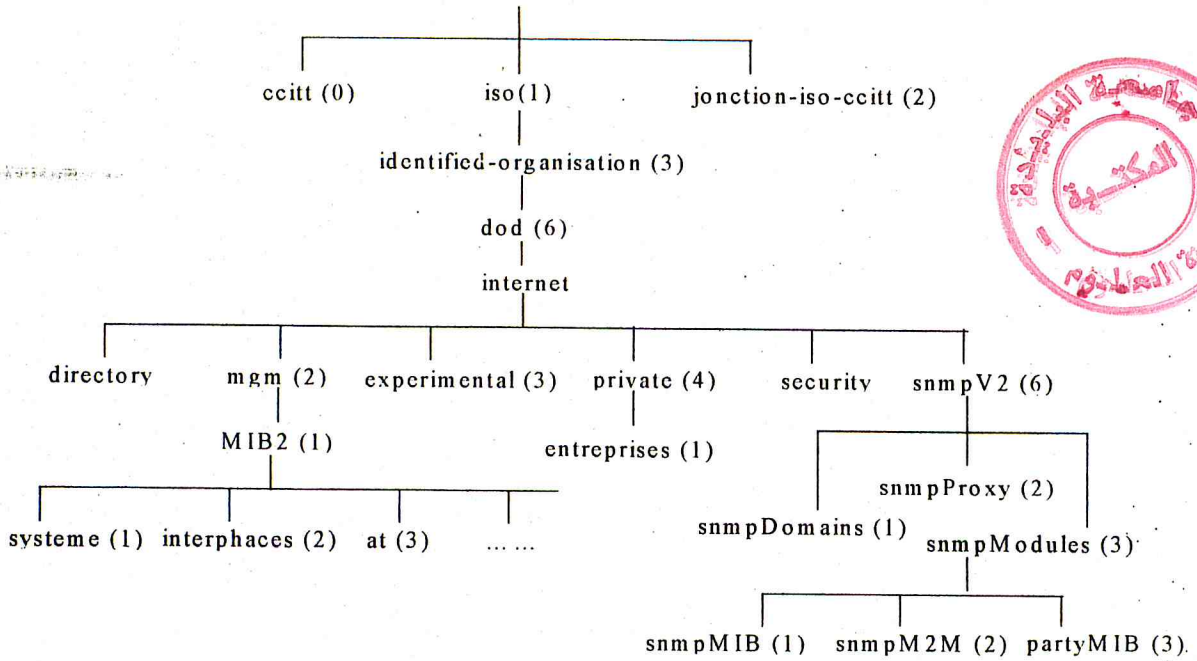


Figure 17 : Les identificateurs d'objets de SNMPv2

La MIB2 de SNMPv1 existe toujours et continuera à être utilisé, et l'augmentation de l'arbre Internet des objets continuera avec des nouvelles définitions du protocole SNMPv2. Pour pouvoir réellement profiter de cette augmentation, SNMPv2 a, amélioré les définitions du SMI (Structure of Management Information), donné une nouvelle définition de comment décrire les nouvelles variables MIB, et aussi amélioré le type des variables dans les définitions ASN1 de la macro OBJECT-TYPE qui améliore la documentation de chaque objet. La création de nouvelles macros avec NOTIFICATION-TYPE qui correspond à la définition de traps ou de "notifications" pour avertir le gestionnaire, MODULE-COMPLIANCE permet de déclarer un ensemble de spécifications pour faciliter l'implémentation d'un ou plusieurs modules de MIB, OBJECT-GROUP qui regroupe un ensemble d'objets à gérer et AGENTCAPABILITIES décrivant les possibilités d'un agent SNMPv2.

3- 2- Les nouvelles commandes de SNMPv2

Il existe dans le protocole SNMPv2 huit PDUs (Protocol Data Unit), mais l'un de ces PDU est amené à remplacer un PDU existant (SNMPv2-Trap), donc il reste que sept actifs:

- **GetRequest, GetNextRequest, Response et SetRequest :** sont les 4 PDUs hérités de SNMPv1.
- **SNMPv2-Trap :** est le nouveau PDU pour émettre des traps, il est conforme d'un point de vue format aux autres PDUs, ce qui simplifie les routines d'interprétation des messages, rendant par la-même le Trap SNMPv1 désuet ("obsolète").
- **GetBulkRequest :** permet de lire un arbre complet de la MIB en une seule requête(on est limité par la taille du message), c'est à dire que cette commande remplace plusieurs GetRequest et GetNextRequest, ce qui a pour effet de supprimer de nombreux messages et donc de diminuer considérablement le volume de données transmis sur le réseau. Cette opération reçoit en retour un message response.
- **InformRequest** PDU est la communication de Gestionnaire à Gestionnaire pour échanger des informations concernant un gestionnaire et pouvant être utile à un autre. **[Lau 96]**

3- 3- Multi-domaines de transports

La reprise des définitions de support de SNMP dans des protocoles différents (OSI, DDP, IPX) que celui de TCP/IP (UDP). SNMPv2 définit formellement l'implémentation du protocole sur ces couches de transports. Ainsi SNMPv2 a différents domaines de transports :

1- Sur UDP dans TCP/IP :

snmpUDPDomain :C'est le transport le plus classique, il permet une certaine forme de compatibilité avec SNMPv1. La version2 continuera à écouter sur le port UDP 161 (Get,GetNext, Set, Response) et à notifier sur le port UDP 162 (SNMPv2-Trap).

2- Sur OSI :

Les messages SNMPv2 utilisent un TSDU (Transport Service Data Unit) pour les ConnectionLess mode Transport Service(CLTS). Mais les modes avec ou sans connexions sont possibles sur une logique OSI (CLNS ConnexionLess mode Network Service ou CONS Connexionb-Oriented Network Service).

3- Sur AppleTalk :

snmpDDPDomain :Les messages sont envoyés à travers un Datagram Delivery Protocol (DDP), qui est utilisé avec le type 8, et l'entité SNMPv2 agit dans son rôle d'agent avec le numéro 8 de socket DDP, alors que c'est le numéro 9 qui est utilisé pour exécuter les notifications (traps).

4- Sur IPX dans Novell:

snmpIPXDomain : Le SNMPv2 est sérialisé dans des IPX datagrams, il utilise les paquets de type 4. Son rôle d'agent est à l'écoute sur les sockets IPX de numéro 36879 (900FH) et envoie les notifications sur le numéro de socket IPX 36880 (9010H).

3- 4- Sécurité

Le principe de « community name » était la seule sécurité existante en SNMPv1, c'est à dire un mot de passe en clair dans chaque message.

1- Concepts de base de sécurité :

Pour pouvoir considérer un système comme sûr, il faut avoir :

- **La confidentialité** (secrecy) : les informations dans un message doivent être accessibles uniquement aux entités autorisées.
- **L'authentification** (Authenticity) : l'origine d'un message doit être identifiée, avec une assurance que l'origine soit correcte.
- **L'intégrité** (integrity) : les informations et ressources ne sont modifiables que par des entités autorisées.
- **Disponibilité** (availablility) : les ressources d'un système doivent être accessibles lorsqu'une entité autorisée en a besoin.

2- Les solutions apportées par SNMPv2 contre les menaces:

SNMPv2 a été conçu pour éviter les menaces suivantes :

- **Mascarade** :Une entité prend l'identité d'une entité autorisée, la solution est : l'algorithme d'authentification (MD5 :Digest Authentification Protocol : permet d'identifier l'émetteur, il est chargé de l'intégrité des messages et de leurs origines).
- **Modifications** des informations : Une entité modifie un message en cours de route (y compris la falsification d'une valeur d'un objet) la solution est : l'algorithme d'authentification.
- **Modification** de l'ordre des messages: Ça concerne le reordonnement, le retard ou la retransmission des messages, la solution est : synchronisation des horloges, timestamps.
- **Divulgation** des informations :Une entité peut observer un dialogue entre un superviseur et un agent (exemple; découverte des mots de passe suite à SetRequest sur userPassword), la

solution est de faire le cryptage (DES : Data Encryption Standard : il s'occupe de l'enchiffrage du message pour le protéger contre la divulgation).

3- 5- Dialogue Gestionnaire à Gestionnaire

Une des améliorations les plus importantes de SNMPv2 est la possibilité de configurer une entité SNMPv2 qui joue le rôle d'agent ou de gestionnaire suivant les tâches qu'on lui demande. Ce qui permet de gérer des configurations importantes où le rôle de Gestionnaire est nécessaire.
L'entité hybride jouant tantôt le rôle de Gestionnaire et tantôt le rôle d'agent se fera par la réception de demandes (messages) émanant tantôt d'agents et tantôt de gestionnaires.

4- Cohabitation SNMP et SNMPv2

L'une des fonctions les plus importantes est la coexistence entre l'ancien SNMP et SNMPv2. Les différents éléments du réseau (HUB, Pont, Routeur...) répondant à SNMPv1, ne sont pas obligés de migrer d'un seul coup vers le nouveau protocole SNMPv2, alors que les nouveaux matériels répondent à SNMPv2. Le fonctionnement en parallèle des deux protocoles a été prévu et une technique de **Proxy** Agent a été élaborée pour cette cohabitation. La fonction de PROXY au niveau du gestionnaire permet de passer d'un protocole SNMPv1 à un autre SNMPv2, c'est à dire faire la traduction. Donc le gestionnaire peut réagir en gestionnaire SNMPv1 ou SNMPv2.

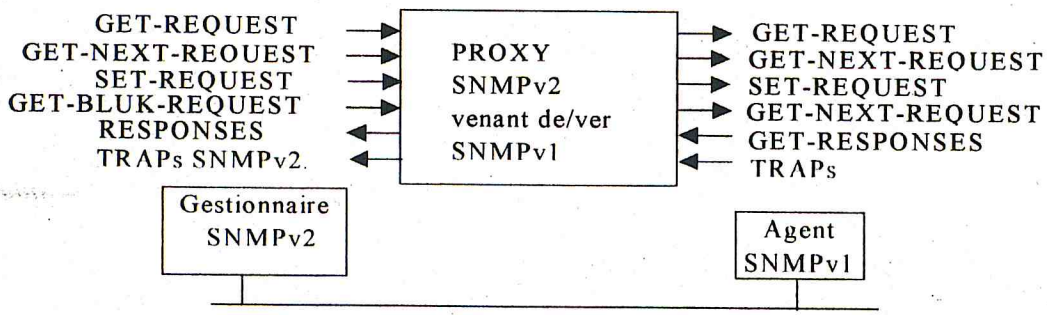


Figure 18 : Cohabitation SNMPv1 et SNMPv2

Les primitives SNMPv2 provenant d'un manager SNMPv2 sont converties dans le format SNMP PDUs pour être envoyées à un agent SNMP suivant ces règles :
- Les primitives get-request, get-next-request et set-request, sont transmises sans modifications,
- Les primitives get-bluk-request sont converties get-nextrequest, avec les même variables.

Les primitives provenant d'un agent SNMP sont converties en primitives SNMPv2 comme suit :
- Les primitives get-responses passent inchangées.
- Les primitives Traps sont converties en primitives Traps SNMPv2.

IXX - Les nouveautés de SNMPv3

La version 3 de SNMP, définie dans les RFC 2570 et 2574, vise à pallier les déficiences de la version 2 afin d'utiliser SNMP de façon sécurisée de bout en bout. Pour cela, il fallait redéfinir l'authentification et le cryptage, les autorisations et les contrôles d'accès et y adjoindre la possibilité d'administrer ces fonctions à distance.

Pour éviter la falsification des messages de commande, en utilisant des mots de passe personnels chiffrés, est né le modèle USM (User-based Security Model). Pour identifier les utilisateurs ou les moteurs SNMP, le modèle USM se sert d'un chiffrement symétrique à clé privée utilisant l'algorithme HMAC-MD5-96. Lors de la transmission d'un message, USM modifie à chaque noeud la clé de chiffrement, ce qui évite, si un segment est endommagé, de menacer l'intégrité des informations reçues par le reste du réseau.
La version 3 de SNMP se révèle complète et sécurisée, elle recourt à un système de chiffrement à clé privée pour sécuriser la transmission des messages sur le réseau. Une parade contre les pirates. Le sous-système de sécurité met plusieurs modèles à disposition. Les requêtes sont réparties vers les générateurs et le receveur de notification et d'applications, et le générateur de commandes. En plus elle interagit avec les anciennes versions du protocole

XX- L'architecture du protocole SNMPv3

Le système de gestion SNMP se compose de :
- **Les agents** :se sont des entités SNMP, ils contiennent les applications :Command Responder qui répend à des commandes et Notification Originator qui envoie des notofications.
- **Le manager** : au moins un, c'est une entité SNMP, il contient les applications : Command Generator qui génère des commandes et Notification Receiver qui reçoit des notifications. Le manager gère et contrôle les éléments de gestion (routeurs, hostes, serveurs terminaux, etc... qui sont contrôlés et commandés à travers l'accès à leurs informations de gestion).
- **Un protocole** de gestion qui sert à transporter les informations de gestion entre les entités SNMP. **[RFC 2571]**

1- L'entité SNMP

Une entité SNMP contient : un moteur SNMP et une ou plusieurs applications associées.

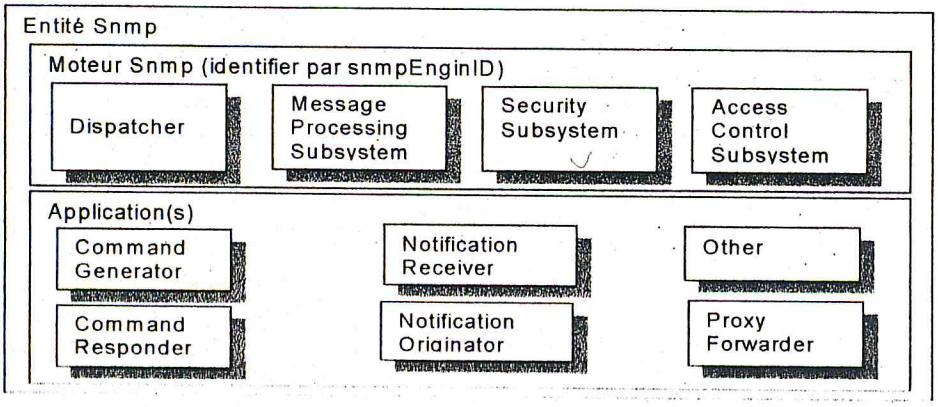


Figure 19 : L'architecture générale d'une entité SNMP

1-1- Moteur SNMP

Dans le domaine d'administration, il est identifié par snmpEngineID; qui peut être le même pour des entités qui appartiennent à des domaines d'administrations différents.
Le moteur SNMP fournit des services pour envoyer et recevoir des messages, contrôler l'accès, authentifier et crypter des messages. Il se compose d'un :
- Transporteur (Dispatcher)
- Module de traitement (Message Processing Subsystem)
- Module de sécurité (Security Subsystem)

- Module de contrôle d'accès (Access Control Subsystem)

1- Dispatcher :

Il est unique, il envoie des messages vers le réseau ou recevoir ceux qui proviennent du réseau, il détermine la version du message pour l 'envoyer au module de traitement correspondant, il fournit une interface abstraite (par des primitives) aux applications pour leur délivrer la PDU ou pour leur permettre d'envoyer des PDU à d'autres entités SNMP distantes.

2- Message Processing Subsystem :

Fait le décodage des paquets qui proviennent du réseau et l'encodage de ceux qui partent dans le réseau. Il permet des changements futur, chaque paquet est marqué d'un numéro de version qui permet au Dispatcher de déterminer le modèle de traitement approprié. En générale chaque modèle de traitement prend en charge une version. Le modèle de traitement est modulaire, ceci permet l'adaptation avec les autres versions. L'ajout d'un nouveau module de traitement permet au moteur de comprendre des messages SNMPng (Snmp Next Generation), ceci permet l'évolution du protocole SNMP sans entraîner la reconstruction complète du moteur. Il est possible de faire fonctionner plusieurs modules de traitements simultanément.

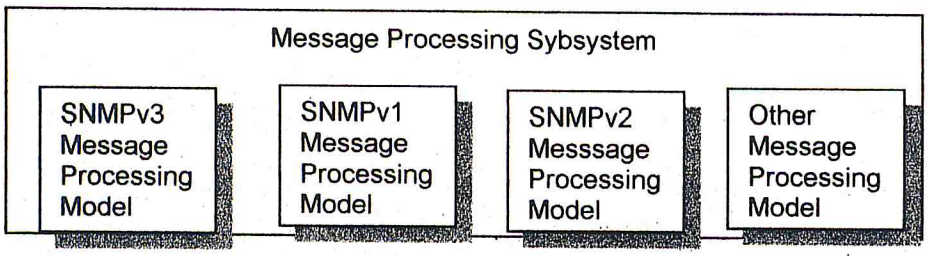


Figure 20 : Module de traitement

3- Security Subsystem :

La sécurité est appliquée dans : la transmission, la réception du message et dans le traitement du contenu du message. Il existe trois fonctions de sécurité : l'authentification, le cryptage et la vérification du temps. Le module de sécurité comporte plusieurs modèles, comme le montre le schéma suivent :

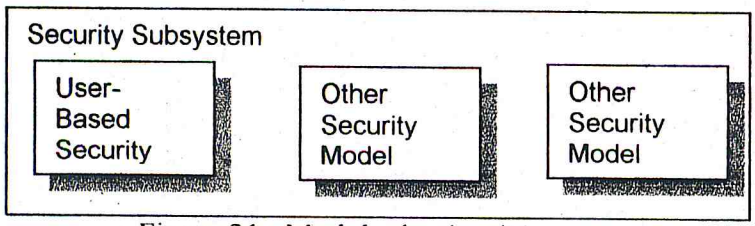


Figure 21 : Module de sécurité

Le modèle de sécurité indique les menaces contre lesquelles il se protége, le but de ces services et le protocole de sécurité utilisé. Le protocole de sécurité indique les mécanismes, les procédures et les objets de la MIB employés pour fournir ces services.

4- Access Control Subsystem :

Chaque application a le choix d'accepter ou de rejeter une requête, si plusieurs requêtes fonctionnent sur le même agent, il serait avantageux de centraliser le contrôle d'accès. On peut faire une seule configuration des droits d'accès. Ce module peut autoriser une requête sur les critères de qui l'a demandé, quel type de requête et quelle information est touchée par la requête.

Chacun d'eux définit une fonction particulière de décision d'accès.

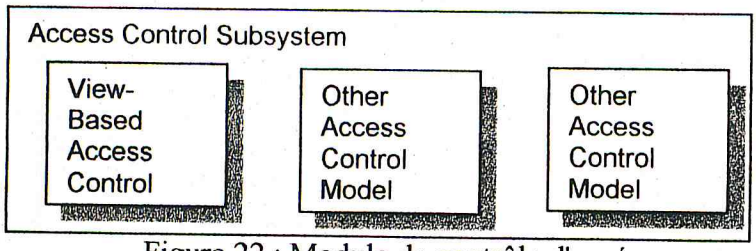


Figure 22 : Module de contrôle d'accés

1-2- Applications

Les applications sont des processus qui interagissent avec le moteur en utilisant des messages qui peuvent être définis dans le protocole ou des messages décrits par une mise en œuvre spécifique.
Les applications qu'on peut trouver sont :
- Command Generator (génère des commandes) : elle gère et manipule les données de gestion, en utilisant des commandes de type Read-Class et/ou de Write-Class.
- Command Responder (répond à des commandes) : permet l'accès aux données de gestion quand elle reçoit une demande de type Read-Class et/ou Write-Class, tout en vérifiant le ContextEngineID qu'il soit égale au moteur récepteur.
- Notification Originator (génère des notifications) : elle surveille un système pour des événements ou des conditions particulières, elle génère des messages de type notification-class. Elle est dotée d'un mécanisme pour déterminer la destination des messages et des paramètres (sécurité, version de protocole) utilisés pour les envoyés. Elle peut être de type Confirmed-Class ou Unconfirmed-Class.
- Notification Receivers (reçoit les notifications) : elle écoute les messages de notifications (messages asynchrones) et génère une réponse quand le message reçu contient une PDU de type Confirmed-Class.
- Proxy Forwwarder : expédie les messages vers les moteurs SNMP selon le ContextEngineID et indépendamment des types d'objets contrôlés et fait suivre la réponse des messages déjà envoyés au moteur duquel le message original a été reçu.

1- L'enregistrement des applications :

Les applications doivent s'inscrire auprès de PDU Dispatcher.

- L'enregistrement de l'application se fait par la primitive RegisterContextEngineID().
- Les paramètres d'enregistrement (ContextEngineID et type PDU) sont examinés pour assurer leur validité, une indication d'erreur (paramètre invalide) est retournée à l'application s'ils ne le sont pas.
- Une indication d'erreur est retournée à l'application, si une autre application s'est déjà inscrite à la combinaison (ContextEngineID, typePDU), car une seule application peut s'enregistrer pour une seule combinaison donnée.

2- Fin d'enregistrement d'une application :

Les applications demandent au PDU Dispatcher de mettre fin à leur enregistrement, en utilisant la primitive UnregisterContextEngineID(). La commande est ignorée si la combinaison (ContextEngineID, typePDU) n'a pas eu lieu.

Architecture d'un manager SNMPv3 :

La figure suivante montre le fonctionnement et les divisions d'un gestionnaire SNMP, on a des divisions entre le transport, le traitement, la sécurité et les applications.

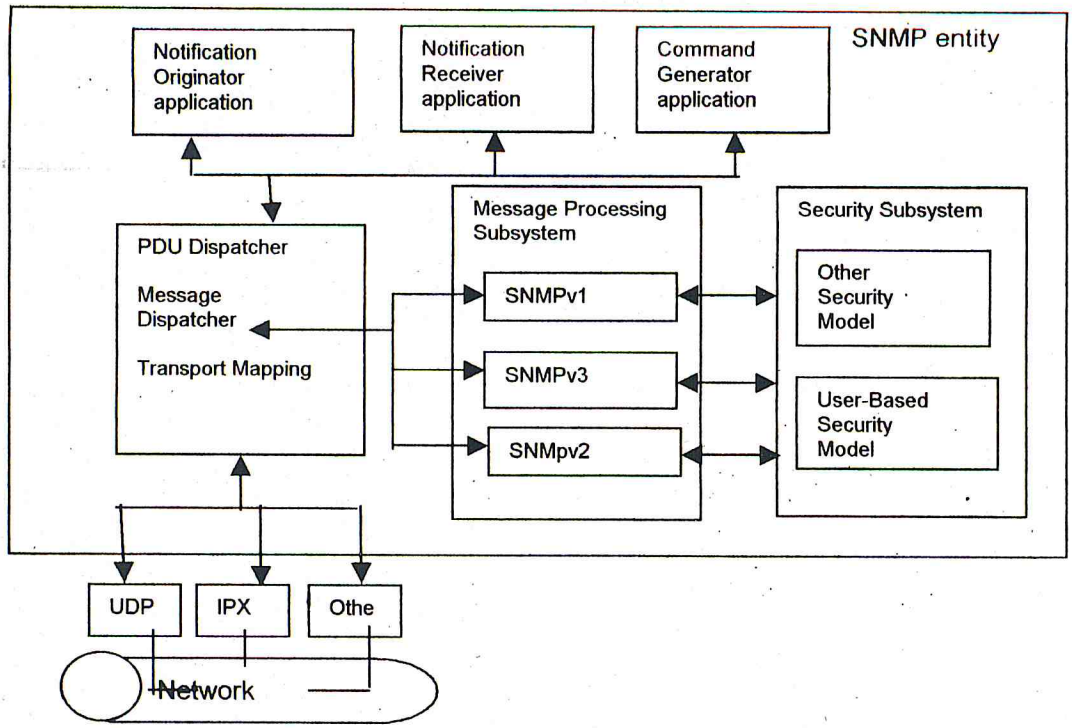


Figure 23: Architecture d'un manager SNMPv3

Architecture d'un agent SNMPv3 :

La figure suivante montre un agent SNMP ou on trouve le module Access Control et les applications qui communiquent avec l'instrumentation.

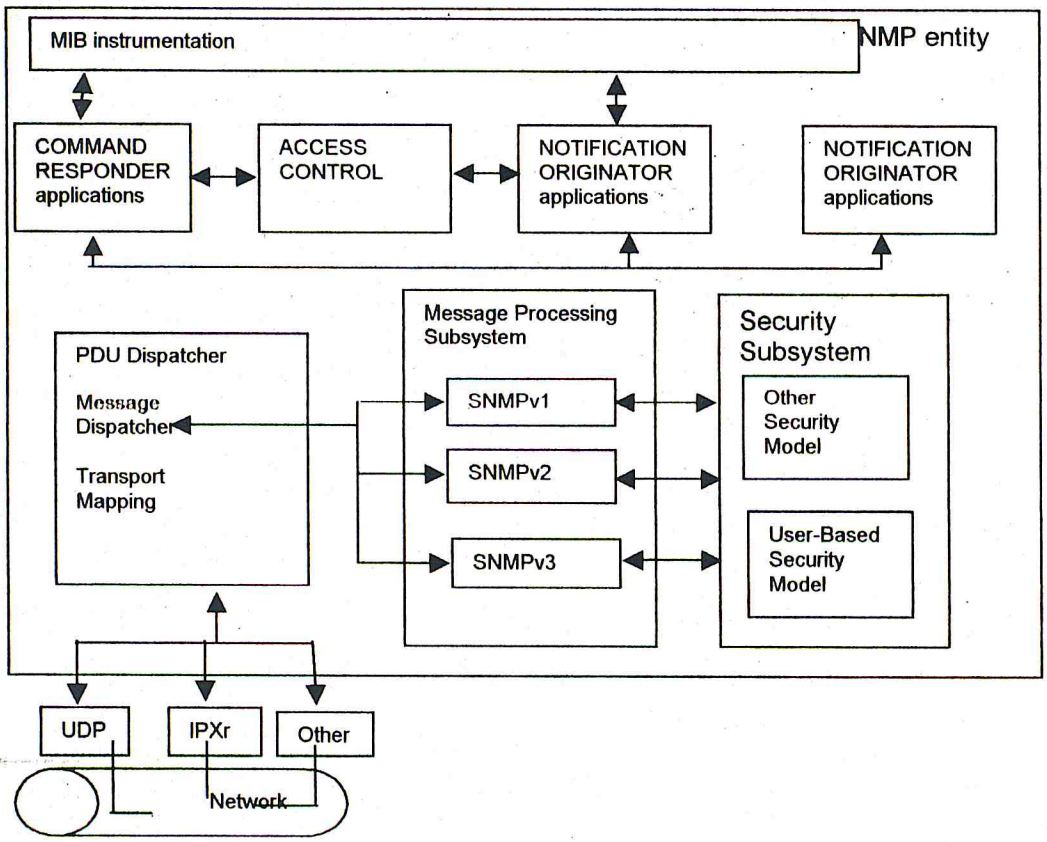


Figure 24 : Architecture d'un agent SNMPv3

2- Le paquet SNMPv3

Comme SNMPv1, SNMPv3 est codé en ASN.1, bien que leur paquets soient très différents. Le numéro de la version est placé au début du paquet pour éviter toute confusion.

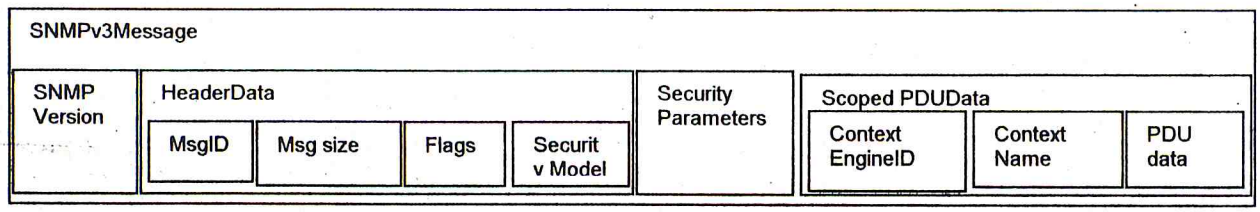


Figure 25 : Description du paquet SNMPv3

Les différents champs du paquet SNMPv3 :
Version SNMP (SNMP version) :
Contient le numéro de version du protocole
Identificateur de message (MsgID) :
Utilisé entre deux entités pour coordonner les réponses associées à des requêtes, les paquets qui sont émis en réponse à une requête portent la même identification que le paquet de la requête.
Taille maximale (Msg Size) :
Détermine la taille maximale que peut accepter un expéditeur quand un autre moteur lui envoie un message (réponse ou autre).
Drapeaux (Flags):
Contient huit bits qui commandent le traitement d'un message, pour l'instant trois bits seulement sont utilisés. Le drapeau SNMPv3 est représenté comme suit :

0	0	0	0	0	Reportable Flag =1	Privacy Flag =1	Authentification Flag =1

Les quatre bits à zéro sont non utilisés.

- Si le champ Reportable Flag est à 1, une réponse doit être retournée par le récepteur du message (exemple PDU Get Request), dans le cas contraire (PDU Trap ou Response).

Les deux bits Privacy Flag et Authentification Flag indiquent au récepteur le niveau de sécurité qui a été utilisé et qui doit être respecté en cas de réponse attendue. Il existe trois niveaux de sécurité, du plus bas au plus haut :
- message non authentifié et non sécurisé
- message authentifié et non sécurisé
- message authentifié et sécurisé

- Si Authentification Flag est à 1, donc le modèle de sécurité utilisé par le moteur qui a envoyé le message doit identifier le nom de sécurité (SecurityName) dans le message et fournir des données suffisantes pour que le récepteur puisse authentifier cette identification, en générale, l'authentification sert à vérifier si :
  - un message n'a pas été ré-orienté.
  - un message n'a pas été modifié en transitant dans le réseau.
  - un message n'a pas été réutilisé.
- Si Authentification Flag est à zéro, SecurityName doit être identifiée par l'expéditeur mais il n'est pas obligé d'envoyer les données, il n'y a rien à authentifier.

- Si Privacy Flag est à 1 alors le modèle de sécurité employé par le moteur qui a envoyé le message, doit aussi protéger la PDU contre la divulgation d'informations, ça veut dire qu'il doit crypter la PDU. Dans le cas contraire la PDU n'est pas protégé.

Remarque : Si Privacy Flag est choisi (mise à 1) alors Authentification Flag doit être choisi.
Les niveaux de sécurité possibles :

Authentification Flag	Privacy Flag	Niveau de sécurité
0	0	Non sécurisé et non authentifié
0	1	Invalide format
1	0	Authentifier mais pas sécurisé
1	1	Authentifier et sécurisé

Tableau 5 : Les niveaux de sécurité possibles

Modèle de sécurité (Security Model) :
Il identifie le type de sécurité utilisé par l'expéditeur des messages et il doit être respecté. Cet identificateur doit identifier de façon unique chaque modèle de sécurité. L'algorithme de cryptage DES (Data Encryption Standard) et l'algorithme d'authentification MD5 (Message Digest Version5) ont été choisis comme algorithmes utilisés dans SNMP.
Les paramètres de sécurité (Security Paramaters) :
Utilisés pour la communication entre les modules de sécurité, dans les moteurs d'envoi et de réception. Ces paramètres changent d'un module de sécurité à un autre.
ScopedPduData :
Représente le ContextEngineID, ContextName et la PDU. Si le bit Privacy Flaf est à un, le ScopedPdu est crypté, dans le cas contraire ScopedPdu n'est pas crypté.
Les identificateurs de contextes (ContextEngineID, ContextName) :
Certains équipements peuvent avoir plusieurs fois les mêmes variables donc plusieurs bases d'informations (MIBs) par agent, un commutateur ATM a plusieurs cartes qui peuvent avoir chacune leurs propres bases d'informations. Le contexte permet de distinguer entre plusieurs bases d'informations et même plusieurs agents.
La PDU :
La PDU contient les variables de la requêtes ou les valeurs de la réponse, et précise quelle opération on désire effectuer, la PDU est décrite dans le RFC 1905.
Chaque PDU appartient à une ou plusieurs classes de PDU comme suit :
Read-Class : contient les opérations de protocole qui recherche l'information de gestion, par exemple dans l'RFC 1905, les opérations GetRequest, GetNextRequest et GetBulkRequest.
Write Class : contient les opérations de protocole qui modifie l'information de gestion, par exemple dans l'RFC 1905, l'opération SetRequest.
Response Class : contient les opérations de protocole qui sont envoyées en réponse à une demande précédente, par exemple dans l'RFC 1905, les opérations Response, Report.
Notification Class : contient les opérations de protocole qui envoient une notification à l'application Notification Receiver, par exemple dans l'RFC 1905, les opérations Trapv2, InformRequest.
Internal Class : contient les opérations de protocole qui sont échangées entre les moteurs de SNMP, par exemple dans l'RFC 1905, les opérations Report-PDU.
Si une réponse est prévue, il sera utile de classifier les PDUs comme suit :
Confirmed Class : contient les opérations de protocole qui font envoyer au moteur une réponse, par exemple dans l'RFC 1905, les opérations GetRequest, GetNextRequest, GetBulkRequest, SetRequest et InformRequest.
Unconfirmed Class:contient les opérations de protocole qui ne renvoient pas des réponses à un moteur donné, par exemple dans l'RFC 1905, les opérations Report, Trapv2 et GetResponse.

3- Les primitives d'appels entre les modules

En plus de la division des modules, SNMPv3 décrit aussi les primitives d'appels de services entre eux.

PDU Dispatcher :
A travers l'expéditeur de PDU (Pdu Ddispatcher), le module de transport fournit des services aux applications. Le tableau suivant montre les primitives de l'expéditeur PDU :

Primitives	Signification
SendPdu()	Permettre à une application d'envoyer une requête ou une notification vers une autre entité SNMP.
ProcessPdu()	Expédier une requête ou une notification entrante à une application.
ReturnResponse Pdu()	Générer une réponse sortante.
ProcessResponse Pdu()	Expédier une réponse entrante à une application.
RegisterContextEngineID()	Permettre à une application d'enregistrer le ContextEngineID().
UnregisterContextEngineID()	Fin d'enregistrement d'une application.

Tableau 6 : Les primitives de l'expéditeur PDU

Module de traitement :
Le tableau suivant présente les primitives du module de traitement et leurs significations:

Primitive	Signification
PrepareOutgoingMessage()	Préparer une requête ou une notification pour l'envoyer vers le réseau.
PrepareResponseMessage()	Préparer une réponse pour l'envoyer sur le réseau.
PrepareDataElement()	Préparer les données d'un message provenant du réseau.

Tableau 7 : Les primitives du module de traitement

Module de sécurité :
Le tableau suivant présente les primitives du module de sécurité et leurs significations :

Primitive	Signification
GenerateRequestMsg()	Générer une requête ou une notification.
ProcessIncomingMsg()	Traiter un message entrant.
GenerateResponseMsg()	Générer une réponse.

Tableau8 : Les primitives du module de sécurité

Module de contrôle d'accès :
Il existe une seule primitive IsAccessAllowed(), elle vérifie la permission d'accès. Ce module travail avec les applications uniquement.

Diagramme de sénario :

Il y a deux diagrammes, le premier nous montre le processus d'envoie de message par Command Generator ou Notification Originator vers le réseau et comment la réponse est retournée.

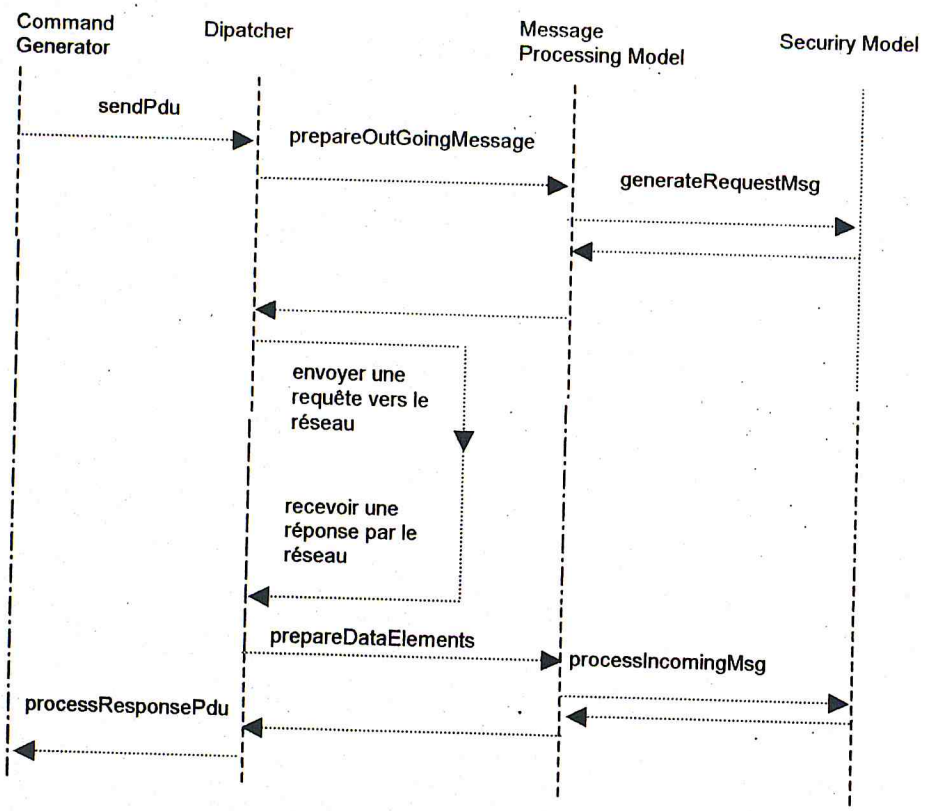


Figure 26: scénario des messages du Command Generator

Le deuxième montre comment Command Responder ou Notification Receiver enregistrent le ContextEngineID, reçoient le message provenant du réseau et comment rendre la réponse.

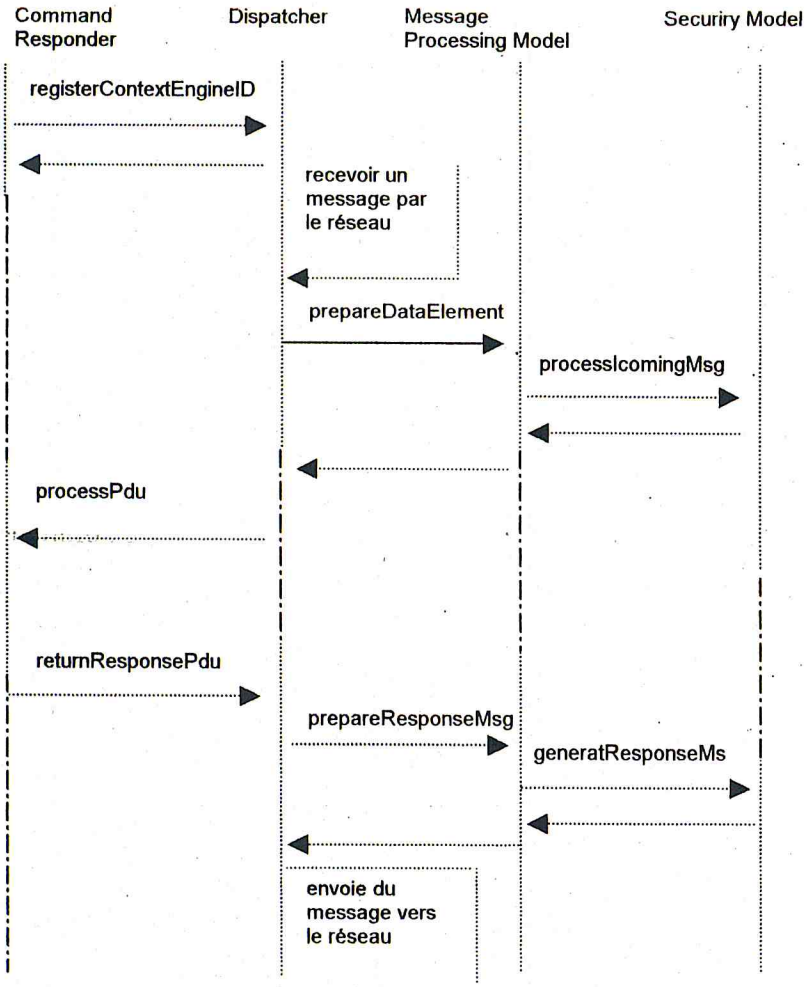


Figure 27 : Scénario des messages du command Responder

XXI- Le traitement et le transport des messages

Le module de transport doit connaître quelques éléments pour pouvoir traiter et transporter un message à travers le réseau comme :

La version du PDU : c'est une valeur qui présente une version du protocole.

Le type de PDU : correspond à la version du PDU contenue dans le message, il présente une opération particulière (GetBulk pour SNMPv2 par exemple).

Le modèle de traitement des messages : décrit des procédures d'une version particulière pour retirer les données contenues dans les messages, produit des messages et demande au modèle de sécurité d'appliquer ses services.

SendPduHandle : raccorde le traitement d'une requête à une réponse entre un moteur et une application.

1- Envoie d'un message vers le réseau (message sortant)

- S'il s'agit d'une requête ou une notification, le moteur procède comme suit :

- Lorsqu'une application veut envoyer au PDUDispatcher un message du genre requête ou trap, elle utilise la primitive SendPDU() .
- Le Dispatcher retourne une indication d'erreur à cette application s'il ne reconnaît pas la valeur du modèle de traitement indiquée dans la primitive.

- Si le Dispatcher reconnaît la valeur du modèle de traitement indiquée dans la primitive, il génère un SendPDUHandle.
- Le Dispatcher envoie le message avec la primitive PrepareOutgoingMessage() au module de traitement selon la version spécifiée dans ce message.
- Si StatuInformation indiqué dans la primitive retourne une erreur alors l'indication d'erreur est retournée à l'application.
- En cas de succès, le SendPDUHandle est retourné à l'application. Maintenant le Dispatcher peut envoyer le message sur le réseau.
  - S'il s'agit d'une réponse le moteur procède comme suit :

-La réponse est envoyée par l'application avec la primitive ReturnResponsePDU() au Dispatcher.
- le Dispatcher envoie la réponse reçue au bon module de traitement avec la primitive PrepareOutgoingMessage().
- Une indication d'erreur est retournée à l'application si le résultat contenu dans la primitive indique une erreur.
- dans le cas contraire, le message est expédié à travers le Dispatcher vers le réseau.

2- Recevoir un message du réseau

Le moteur SNMP procède comme suit :
- L'incrémentation du SnmpInPkts counter
- La détermination de la version du message, dans le cas ou elle échoue SnmpInASNParseErrs counter est incrémenté.
- L'origine du domaine de l'expéditeur et du destinataire est déterminée.
- La primitive PrepareDataElements() expédie le message vers le bon modèle de traitement.
- Si le résultat contenu dans cette primitive indique une erreur, le message est rejeté.
- Dans le cas de succès, les données sont préparées à l'envoie comme suit :
  - Expédier des PDUs pour des messages entrants

Ça dépend de la valeur de SendPduHandle. Si elle rend une valeur donc c'est une réponse rentrante alors on suit les étapes suivantes :
- SendPduHandle est employé pour déterminer quelle application attend cette réponse.
- Le message est abandonné si aucune application n'est trouvée.
- La réponse est expédiée à l'application destinataire avec la primitive ProcessResponsePdu() dans le cas contraire.

Si SendPduHandle ne retourne aucune valeur donc c'est une requête (demande) ou une notification alors on suit les étapes suivantes :
- Pour déterminer quelle application a été enregistrée pour cette requête ou notification, on emploie la combinaison (ContextEngineID, PduType).
- S'il n'y a aucune application inscrite alors :
  - Incrémentation de SnmpUnKnownPduHandle counter.
  - Un message de réponse est produit avec PrepareResponseMessage().
  - Si le résultat contenu dans cette primitive ne retourne pas une erreur alors la réponse est envoyée à celui qui a fait la requête.
- La prémitive SendPdu() expédie la PDU à l'application déstinataire.

XXII- La sécurité dans SNMPv3

Comme SNMPv2, SNMPv3 lutte contre les menaces (de mascarade, de divulgation de l'information, de modification d'information, de modification d'une suite de message) pour lesquelles le modèle de sécurité assure la protection.
Donc le but du modèle de sécurité est de lutter contre ces menaces.

1- Le modèle de sécurité User-Based-Security :

Quatre mécanismes de sécurité sont utilisés dans SNMPv3 par UserSecurityModel. Chacun d'eux a pour but d'empêcher un type d'attaque.

L'authentification :
Empêche le changement du paquet SNMPv3 en cours de route et valide le mot de passe de l'usager qui transmet la requête.
La localisation des mots de passes :
Empêche quelqu'un de compromettre la sécurité d'un domaine d'administration, même si la sécurité d'un des agents du domaine est compromise.
Le cryptage :
Empêche la lecture des informations de gestion contenues dans un paquet SNMPv3.
L'estampillage du temps :
Empêche la réutilisation d'un paquet SNMPv3 valide que quelqu'un a déjà transmis. **[Yli 88]**

Ces mécanismes utilisent des mots de passes Partagés (shared passwords), connu par la station de gestion et l'agent seulement. Il existe des mots de passes Public, mais ils ne sont pas utilisés dans SNMPv3. De préférence l'administrateur place le mot de passe sans qu'il passe par le réseau s'il a un accès physique à la station et à l'agent. L'agent a une interface limitée avec l'utilisateur (pas d'écran, pas de clavier) donc la façon de configurer un agent est d'utiliser SNMP.

1-1- L'authentification

Son rôle est d'assurer qu'un paquet ne change pas pendant la transmission et que le mot de passe est valide pour l'utilisateur qui fait la requête.
Pour assurer son rôle, elle est dotée des fonctions de hachage à une seule direction tel que MD5 et SHA-1, elles prennent en entrée une chaîne de caractères de longueur indéfinie, et génèrent en sortie une chaîne d'octets de longueur finie (16 octets pour MD5 et 20 octets pour SHA-1).

Propriétés des fonctions de hachage à une seule direction :
Il est très difficile de trouver deux chaînes de caractères qui passent dans une fonction de hachage à une direction, nous donnent le même résultat. Si on trouve deux chaînes qui ont le même code de hachage, on dit que l'on a trouvé une collision.
Ou encore :
Etant donné une chaîne d'octets qui est le résultat d'une fonction de hachage à une direction.
Il doit être très difficile de trouver une quelconque chaîne d'entrée qui, une fois passée dans la fonction, donne cette même chaîne en sortie.
Remarque :
Les chances que deux chaînes de caractères aient le même résultat après être passé par la fonction de hachage à une direction (SHA-1) est de 1 sur 10^{48}. Même avec des ordinateurs très puissants et dans un temps logique, il est impossible de trouver une combinaison pareille.

Le résultat de la fonction de hachage est placé dans le bloc des « paramètres de sécurité » du paquet SNMPv3.
Pour transmettre une information, on doit utiliser un mot de passe« Partagé » qui soit connu par le transmetteur et le receveur uniquement.
Le mot de passe est validé sans qu'il soit transmis par le réseau. Si un paquet SNMPv3 passant par le réseau est saisi, il est très difficile de trouver le mot de passe pour changer son contenu.

Le mécanisme d'authentification :

Le transmetteur ajoute un mot de passe aux informations à transmettre, puis il les passe dans une fonction de hachage à une direction, après, les données et le code de hachage sont transmis sur le réseau.
Le receveur rajoute le mot de passe au bloc de données reçues, puis les fait passer par la fonction de hachage à une direction, enfin la vérification : si le code de hachage est identique à celui transmis, le transmetteur est identifié.

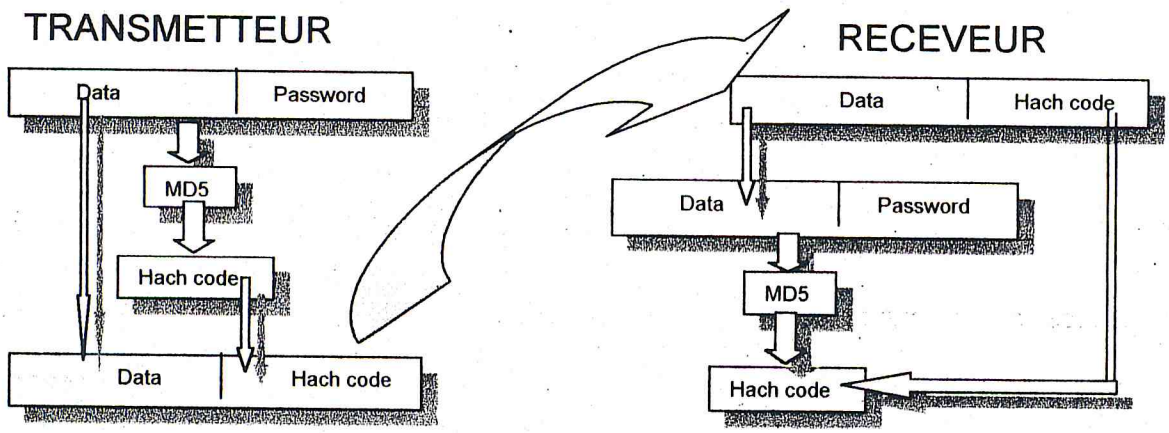


Figure 28 : L'authentification d'un message

1-2- La localisation

Une station de gestion communique avec plusieurs agents. Si ces agents ont le même mot de passe et qu'un seul d'entre eux soit volé ou compromis, alors le mot de passe pourrait être utilisé pour administrer tous les autres agents.
La solution est d'utiliser un seul mot de passe, mais de passer par une étape de « localisation ». Un mot de passe localisé fonctionne qu'avec un seul agent. D'abord il nous faut une chaîne de caractères qui soit unique à chaque agent, pour cela SNMPv3 utilise ContextEngineID. Cette chaîne est générée par un ensemble de données comme : l'adresse MAC de la carte Ethernet, l'adresse IP, des nombres aléatoires ou bien une chaîne spécifiée par l'administrateur. **[RFC2574]**

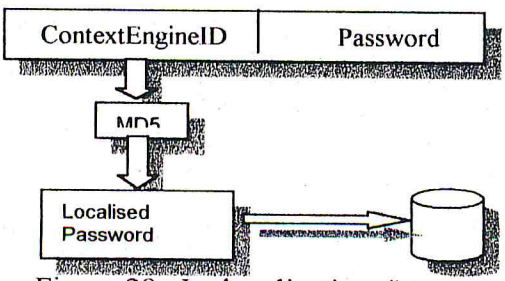


Figure 29 : La localisation d'un mot de passe

Le mécanisme de localisation :
Trouver le ContextEngineID de l'agent auquel on veut envoyer une requête et puis on lui ajoute le mot de passe et on le fait passer dans une fonction de hachage à une direction. C'est ce mot de passe localisé et mémorisé dans l'agent, qui est utilisé dans l'authentification et le cryptage des paquets SNMPv3 par la station de gestion.

1-3- Le cryptage

Le but du cryptage est d'empêcher une personne d'obtenir des informations de gestion en écoutant les requêtes et les réponses d'autres personnes sur le réseau.
Contrairement à l'authentification qui est appliquée à tout le paquet, le cryptage est seulement appliqué sur le Scoped PDU.
Le cryptage se fait par un mot de passe partagé. Mais pour des raison de sécurité, SNMPv3 utilise deux mots de passe différents, un pour l'authentification et un pour le cryptage, pour leur permettre d'être indépendant et aucun d'eux ne peut former un risque pour l'autre.
Pour le cryptage, SNMPv3 utilise DES qui ressemble à une fonction XOR compliquée.

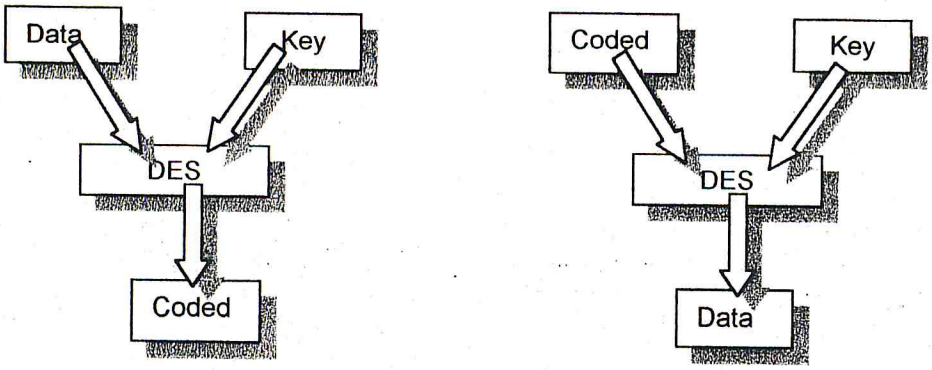


Figure 30 : Le mécanisme de cryptage

Le mécanisme de cryptage:
On utilise une clé de 64 bits. Comme DES crypte 64 bits à la fois et comme les informations qu'on doit crypter sont plus longues, alors on utilise le chaînage de blocs DES de 64 bits.

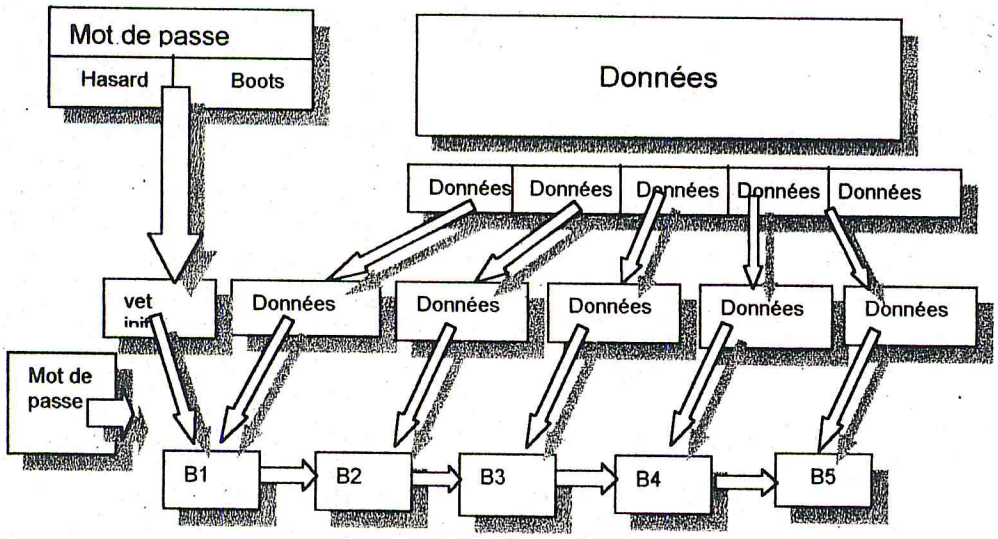


Figure 31 : Le fonctionnement de cryptage pour le chaînage des blocs DES

Une combinaison du mot de passe, d'une chaîne aléatoire et d'autres informations forme un « vecteur d'initialisation ». Chacun des blocs de 64 bits est passé par le DES et est chaîné avec le bloc précédent avec un XOR, le premier bloc est chaîné avec un XOR avec le vecteur d'initialisation.

Remarque :

Le vecteur d'initialisation est transmis avec chaque paquet dans le champ paramètres de sécurité qui fait partie du paquet SNMPv3.

1-4- L'estampillage du temps

Jusqu'ici, les mécanismes d'authentification, de localisation et de cryptage n'empêchent personne de saisir un paquet SNMPv3 du réseau et d'essayer de le retransmettre plus tard, sans modification.

Même si cette personne n'a pas l'autorisation nécessaire, elle envoie le paquet, authentifié et crypté correctement par la station de gestion à l'équipement. Ce genre d'attaque s'appel le « Replay attack ».

Pour éviter cette attaque ; le temps est estampillé sur chaque paquet, à sa réception, on compare le temps actuel avec le temps dans le paquet, si la différence est supérieure à 150 secondes, le paquet est ignoré.

Les agents SNMP sont des micros contrôleurs qui n'ont pas d'horloge avec l'heure courante et précise. Il est difficile de garder toutes les horloges synchronisées.

Puisque SNMPv3 n'utilise pas l'heure normale ; une horloge différente est utilisée dans chaque agent qui, va garder le nombre de secondes depuis la mise en circuit de ce dernier, un compteur est également utilisé pour compter le nombre de fois où l'équipement a été mis en circuit ; ces compteurs sont appelées BOOTS (nombre de fois en circuit) et TIME (nombre de secondes depuis la dernière fois que l'équipement a été mis en circuit).

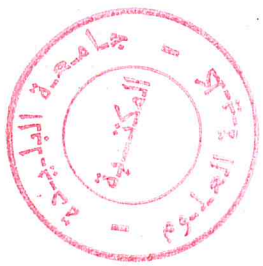
La combinaison du BOOTS et TIME donne une valeur qui augmente toujours, et qui peut être utilisée pour l'estampillage.

Comme chaque agent a sa propre valeur du BOOTS/TIME , la station de gestion doit garder une horloge qui est synchronisée pour chaque agent qu'elle contacte, au contact initiale, la station de gestion obtient la valeur du BOOTS/TIME de l'agent et synchronise une horloge distincte.

XXIII- Conclusion

Dans ce chapitre nous avons vu l'architecture, le fonctionnement, la spécification et le format des messages SNMPv1, nous avons constaté que cette version a un problème de sécurité. Ensuite nous avons présenté la deuxième version bien qu'elle n'a pas été utilisée malgré sa nouveauté, par manque de sécurité.

Nous avons ensuite décrit les détails du protocole SNMPv3 et comment il est découpé en modules. Nous avons également vu comment SNMPv3 assure l'authentification et la confidentialité des messages. Nous avons constaté qu'il est assez modulaire pour permettre des modifications, et qu'il est basé sur des concepts de sécurité bien éprouvés.



Chapitre IV :
Etude conceptuelle

I- Introduction

L'étude conceptuelle est une partie importante dans la réalisation de notre logiciel, nous avons choisi le modèle de conception en cascade.
Le modèle en cascade est le modèle le plus classique, il décrit le cycle de vie d'un logiciel par la succession des étapes suivantes :
La spécification des besoins, la conception, l'implémentation, le test et la validation. **[Bru 01]**

Nous avons vu dans les chapitres précédents que le protocole SNMP offre une importante gestion de réseau facile et simple, SNMP a été développé pour permettre à l'administrateur du réseau, d'interroger et de surveiller les éléments de son réseau sans se déplacer, de modifier leur configuration, faire des tests de sécurité et observer différentes informations liées à l'émission de données.

Le principe de SNMP est très simple, sur chacune des machines, on installe un petit programme : l'agent SNMP. Cet agent enregistre en permanence des informations relatives à la machine. Il stocke ces informations dans une MIB (Managment Information Base), une base de données. Ainsi, de son ordinateur, l'administrateur peut interroger chacune de ses machines et obtenir les informations qu'il souhaite, comme par exemple le nombre d'octets reçus et envoyés. Il peut aussi modifier certaines informations.

Le protocole SNMP fonctionne sur un modèle client-serveur, où il n'y a qu'un seul client, la station d'administration (NMS : Network Managment Station) et beaucoup de serveurs (chaque agent SNMP), le client interroge les serveurs pour récupérer les informations.

Chaque agent est placé sur un nœud du réseau qui est dit administrable (MN : Managed Node). Ces nœuds peuvent être soient des hotes (station de travail ou serveur), soit des éléments d'interconnexion (switchs, hubs, routeurs…), soit des supports physiques (câbles).

L'environnement de gestion de SNMP est constitué de plusieurs composantes : Une station de gestion de réseau (NMS :Network Managment Station), des éléments de réseaux (NE :Network Elements), des variables MIB (Managment Information Base) et un protocole.

Notre travail consiste à développer une application qui :

Permet de réaliser la cartographie du réseau grâce à SNMP, qui se charge de récolter des informations sur les différentes bases d'informations d'éléments du réseau (demande, modification et réponse) et de lancer des alertes (des notifications) en cas de dysfonctionnement.

Permet de capturer les paquets transitant sur le réseau, en utilisant une méthode de conception de capture de trames, qui se résume en trois niveaux ; le premier niveau, c'est le niveau noyau, c'est là où la capture se réalise réellement, puis vient le niveau du module d'interface et de dialogue, qui réalise la communication entre le niveau noyau de capture, et le niveau du module de capture utilisateur qui attend la remonté des paquets capturés, pour d'éventuelles modifications et enfin le dernier niveau de l'architecture, le module de capture utilisateur, c'est le programme de capture de l'utilisateur qui réalisera aussi une analyse sur les paquets capturés pour d'éventuelles traitements.

Permet aux paquets capturés d'être analysés et d'extraire des informations à partir des statistiques obtenues afin de déceler les goulots d'étranglement,

Permet d'améliorer, l'utilisation de la bande passante, et la qualité de service sur l'ensemble des éléments constituants le réseau.

II- Méthode de conception

Pour bien maîtriser la progression de notre travail, nous avons établi une architecture, cette architecture consiste à découper notre projet en différentes parties ou modules, cette décomposition permet de partitionner un système complexe en plusieurs sous systèmes, chacun d'eux exécute une tâche bien précise pour bien cerner chaque problème de conception à part.
Grâce à cette architecture, notre application peut être mise à jour sans entraîner des changements dans tous les modules, elle offre alors la possibilité de faire des interventions séparées sur ces modules.

La figure suivante décrit schématiquement la méthode de conception utilisée pour réaliser les buts de notre projet :

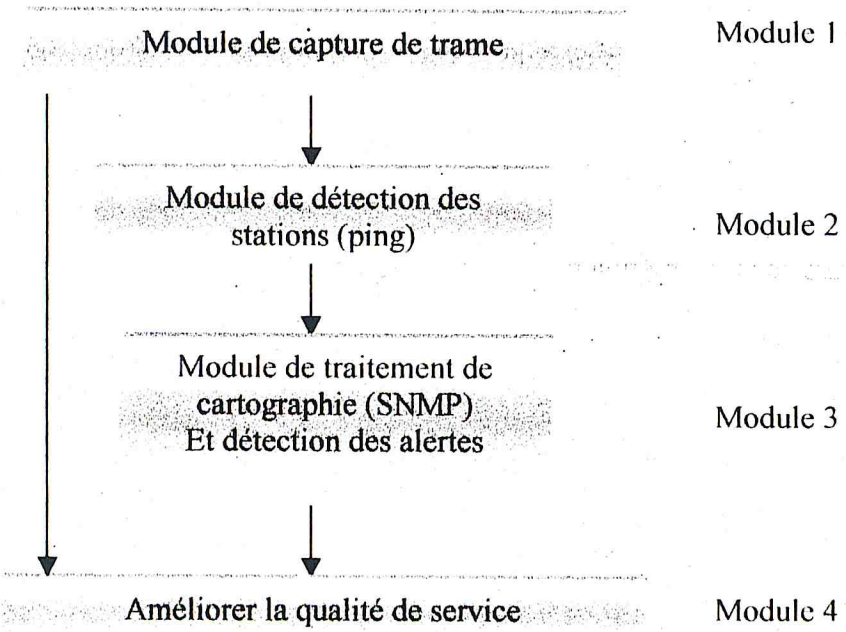


Figure 1 : La méthode de conception utilisée pour réaliser notre projet

Les quatre principales étapes suivantes résument l'idée de base :

- **Le module 1 « capture des trames »** :

Ce module doit réaliser une capture des paquets de données, afin de les analyser et faire des statistiques pour des traitements futurs.

La capture des trames permet d'écouter le trafic et d'extraire les différentes informations circulant dans le réseau, pour ce faire, nous utilisons des primitives et des fonctions pour dialoguer avec le pilote de la capture qui communique à son tour avec le gestionnaire de la carte réseau Ethernet pour arriver au niveau de l'adaptateur (interface réseau) et faire remonter les informations à l'application de l'utilisateur.

Notre architecture de capture de trames et d'analyse de réseaux se constitue de trois niveaux hiérarchiques:

Le NIVEAU 1: le noyau de l'architecture, il représente l'interface vers le réseau local Ethernet. Cette interface est constituée de deux modules qui réalisent le lien et le dialogue avec le bus du réseau.
Le premier module est le gestionnaire de la carte réseau ou bien le pilote de la carte réseau (NIC, pour Network Interface Card).
Le deuxième module est le noyau de l'architecture, il est situé au-dessus du gestionnaire de la carte réseau, il représente le module de la capture ou bien le Pilote de la capture.
Le NIVEAU 2: le module d'Interface et de dialogue, il fait l'interface en réalisant la communication entre le niveau noyau (qui réalise la capture) et le niveau application de l'utilisateur (qui attend la remontée des paquets de données capturés).
Le NIVEAU 3: le Programme de la capture, il représente le programme de capture de l'utilisateur, il réalise des traitements sur les paquets de données capturés et remontés comme : la lecture des paquets capturés, décodage des différents champs constituant les paquets capturés, préparer le fichier de sortie qui va recevoir les paquets capturés, l'affichage des paquets, l'analyse et l'interprétation des paquets, etc.

Dans la section correspondante à ce module, nous aborderons chaque niveau de l'architecture de capture des trames en détail.

- **Le module 2 « détection des stations »** :

Pour connaître l'état des différentes stations du réseau, c'est à dire si une station est active ou pas, il faut lancer un programme « ping » qui scrute ces stations.

C'est un utilitaire destiné à savoir si une machine est connectée ou pas au moment où nous lançons la commande. Il permet également de déterminer le temps de réponse de la machine, c'est à dire le temps que mettent les paquets pour aller à la machine destination et en revenir.

Dans la section correspondante à ce module, nous détaillerons l'algorithme utilisé pour tester les stations (grâce à la fonction du ping).

- **Le module3 « le traitement de la cartographie et de détection des alertes »** :

Parmi les protocoles de gestion de réseau, nous avons opté pour le protocole SNMP (Simple Network Managment Protocol).
SNMP peut être utilisé pour gérer des logiciels et bases de données à distance.

Le protocole SNMP permet de répondre à un grand nombre de besoins : disposer d'une cartographie du réseau, fournir un inventaire précis de chaque machine, mesurer la consommation d'une application, signaler les dysfonctionnements par des alertes.

Dans la section spécifiée à ce module, nous allons montrer, le mécanisme et le principe du fonctionnement de l'agent lors de la réception des demandes envoyées par la NMS, et comment il envoie sa réponse (response), et comment il agit sur les alertes en envoyant des (Traps) à la NMS.
Nous allons monter également le principe du fonctionnement du manager pour l'envoie des requêtes de demande (Get, Get-next) et de modification (Set) et la manière dont il reçoit les réponses et les notifications envoyées par l'agent.

- **Le module 4 « qualité de service »** :

La qualité de service (QoS :Quality of Service en anglais) se résume à désigner une différenciation dans la qualité d'acheminement des données circulant sur le réseau. En d'autre termes, la qualité de service devait tendre à ce que le trafic dit « normal » ait plus de chances de ne pas atteindre sa destination que le trafic « prioritaire ».

Le support de qualité de service (QoS) dans l'Internet reste encore aujourd'hui un verrou technologique important. De nombreux protocoles (IntServ, DiffServ, MPLS, etc.) ont été proposés pour apporter une différenciation de services dans l'Internet.

Aujourd'hui, ce concept couvre un champ beaucoup plus large ; la qualité de service désigne avant tout un transfert à débit garanti entre l'émetteur et le récepteur des données, et ce avec des temps d'attente (de latence) réduits au minimum.

Au moins deux raisons confirment l'importance de ce concept :

La première est que dans les périodes de congestion du réseau, il est primordial de trouver un mécanisme qui autorise un traitement différencié des données en circulation ; on peut parler de trois classes de service :

1. GS :Garanteed Service, c'est le trafic prioritaire sur les noeuds du réseau. Chaque routeur dispose d'un buffer prioritaire sur les autres classes. Il subit peu de pertes et les délais d'acheminement sont très courts.
2. AS : Assured Service, ce type de trafic n'est pas prioritaire, mais il subit statistiquement de faibles pertes dans le réseau. En cas de saturation imminente, rejeter des paquets de type best effort plutôt que ceux de type AS.
3. BE : Best Effort, le reste des paquets, c'est le trafic actuel.
Il s'agit de marquer et de contrôler les paquets au niveau des routeurs à l'entrée du réseau.

La seconde est que par ce biais, les fournisseurs d'accès trouvent le moyen de proposer des services à valeur ajoutée à leurs clients afin de se distinguer de leurs concurrents.

De nombreuses enquêtes montrent que les internautes mettent la fiabilité, la rapidité d'accès, les coûts d'accès et le service à la clientèle en tête de leurs exigences. Les instances chargées de la définition de la qualité de service ont donc un rôle clé à jouer.

Rappel : dans notre projet, le dut n'est pas d'implémenter ou de créer un mécanisme de qualité de service.

III- Capture et analyse des trames

1- Introduction

La capture des trames est un moyen permettant d'écouter le trafic d'un réseau, c'est-à-dire de capturer les informations qui circulent.
En effet dans un réseau non commuté, les données sont envoyées à toutes les machines du réseau. Toutefois, dans une utilisation normale les machines ignorent les paquets qui ne leurs sont pas destinés. Ainsi, en utilisant l'interface réseau dans un mode d'opération appelé libéral (ou aussi le mode Promiscous), dans ce cas l'adaptateur réseau (une carte réseau Ethernet, une carte réseau sans fil, ...) accepte tous les paquets transitant par le segment de réseau, y compris les paquets à destination d'autres nœuds. Les trames interceptées par la carte réseau sont misent dans des buffers afin ; de les décoder et d'extraire les informations nécessaires, et de les analysées.

L'analyse du réseau se fait sur les trames capturées afin d'établir des statistiques sur le flux de données circulant dans le réseau.

Après avoir capturé les trames du réseau, l'analyse intervient pour spécifier (déterminer) par exemple le type d'application utilisé, en établissant des pourcentages pour ; déterminer quelle application occupe plus souvent notre réseau.
L'analyse se fait également par rapport à une machine cible du réseau.
Par exemple : pour chaque poste, voir combien une application particulière est utilisée, ou bien encore, pour une adresse donnée, voir tous les protocoles utilisés.

L'analyse nous aide à faire des statistiques pour d'éventuelles utilisations ; par exemple, dans le cas où on décide d'établir une QoS et d'effectuer des priorités pour les applications circulants sur le réseau.

2- L'algorithme général (partie capture de trame)

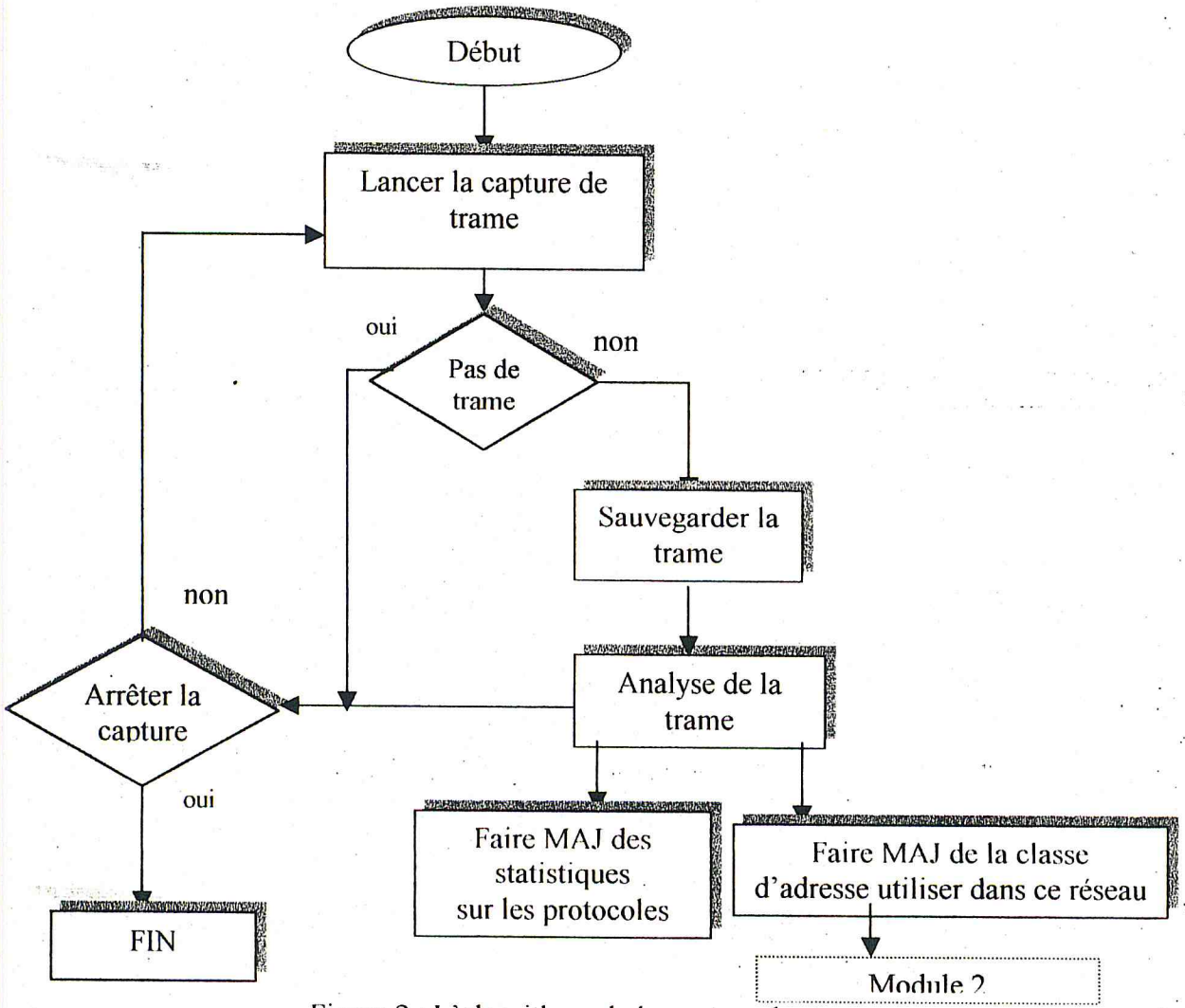


Figure 2 : L'algorithme de la capture des trames

Pour les résultats de cette partie nous avons :

- Les statistiques (pourcentage) des protocoles qui circulent dans le réseau.
- La classe d'adresse utilisée par le réseau, cette classe est utilisée par le module 2 pour la cartographie du réseau.

Remarque : Cette partie ne donne pas l'adresse du réseau seulement mais la classe d'adresse aussi.

3- Méthode de conception

Elle se base sur une technique de conception en modules qui interagissent entre eux. La décomposition en modules permet de diviser un système complexe à concevoir en sous-systèmes dédiés à des tâches précises, ce qui permet de mieux cerner les problèmes de conception, mais aussi d'offrir la possibilité de faire des interventions séparées sur ces modules. La Figure suivante décrit schématiquement la méthode de conception utilisée pour réaliser la capture des trames :

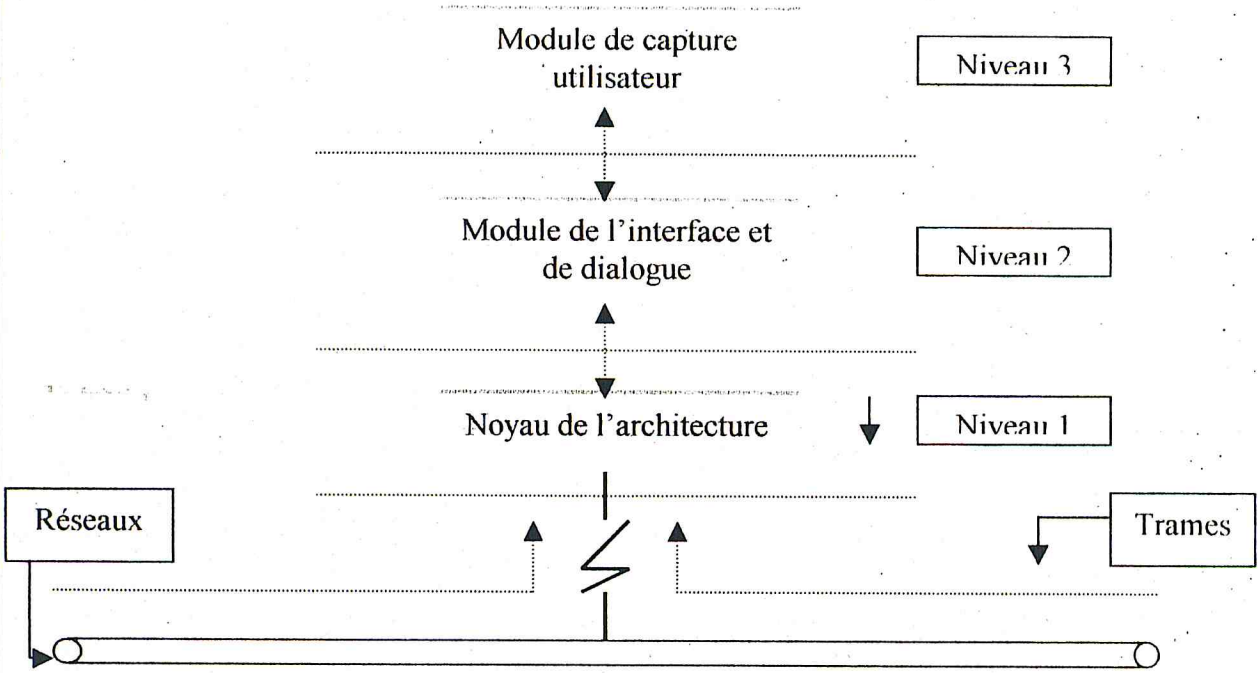


Figure 3 : Architecture de capture des trames (paquets) Ethernet

Les trois principales étapes suivantes résument l'idée de base :

- Les paquets de données qui circulent sur le segment du réseau local Ethernet, sont représentés par le trafic recueilli au niveau de ce réseau.
- Les paquets de données transitent le niveau 1, qui représente le niveau noyau de l'architecture, ceux-ci sont remontés vers les niveaux supérieurs de l'architecture pour des traitements ultérieurs.
- Lorsque les paquets sont traités par les différents modules, les résultats des opérations effectuées sur ces derniers sont dirigés vers les organes de sorties se trouvant au niveau de l'utilisateur (fichier de sortie /monitoring/rapport…).

4- Les principaux niveaux constituant l'architecture de capture

Notre logiciel (partie de la capture des trames) doit réaliser une capture des paquets de données, afin de les analyser et faire des statistiques pour d'éventuels traitements.
Pour cela, trois principaux niveaux hiérarchiques constituent notre architecture de capture de trames et d'analyse de réseaux et interagissent entre eux, ces derniers se présentent comme suit :

4-1- Le Noyau de l'architecture (NIVEAU 1)

Représente le plus bas niveau de l'architecture de la capture des trames, appelé « le noyau de l'architecture » en réalité, ce noyau représente l'interface vers le réseau local Ethernet. En d'autre termes cette interface est constituée de deux principaux modules qui réalisent le lien ainsi que le dialogue avec le segment (le bus du réseau).
Le premier module concerne le gestionnaire de la carte réseau, appelé aussi « pilote de la carte réseau » (NIC, pour Network Interface Card).
Le deuxième module constituant le noyau de l'architecture, représente le module de la capture (Pilote de la capture), situé au-dessus du gestionnaire de la carte réseau.

4-2- Le Module d'interface et de Dialogue (Niveau 2)

Représente le deuxième niveau de l'architecture de la capture, il est situé au-dessus du noyau. Ce niveau est appelé le « Module d'Interface et de dialogue », comme son nom l'indique, ce niveau fait l'interface en réalisant la communication entre le « Niveau noyau » qui réalise effectivement la capture, et le niveau application de l'utilisateur qui attend la remontée des paquets de données capturés.

4-3- Le Programme de la capture (Niveau 3)

Le troisième niveau constituant notre architecture représente le programme de capture de l'utilisateur, qui va réaliser des traitements sur les paquets de données qui ont été capturés et remontés tels que :

- ✓ La lecture des paquets capturés.
- ✓ Décodage des différents champs constituant les paquets capturés.
- ✓ Préparer le fichier de sortie qui va recevoir les paquets capturés.
- ✓ L'affichage des paquets.
- ✓ L'analyse et l'interprétation des paquets.
- ✓ Etc.

Ce niveau dispose d'une librairie de procédures et de fonctions de haut niveau qui peuvent être utilisées dans l'application de capture de trames Ethernet de l'utilisateur. Cette librairie permet à l'utilisateur l'envoi des requêtes au deuxième niveau constituant l'architecture de capture, c'est-à-dire le niveau Module d'interface et de dialogue, pour pouvoir remonter les paquets de données qui circulent dans le segment Ethernet, et qui ont été capturés par le noyau de l'architecture.
En outre, cette librairie dispose d'un ensemble de fonctions permettant de faire des traitements sur les paquets capturés.

Remarque : Les différents traitements et opérations effectués sur les paquets capturés (Tels que : l'affichage, la lecture, le décodage, l'analyse...) sont réalisés indépendamment du hardware du réseau et du système d'exploitation.

5- Structure de la capture

Pour capturer les paquets de données transférés via le réseau, l'application de la capture a besoin de dialoguer directement avec le hardware du réseau. Pour cette raison, le système d'exploitation doit offrir une série de primitives appelées « Primitives de capture » pour communiquer et recevoir directement les données de l'adaptateur du réseau.
L'objectif principal de ces primitives de capture, consiste essentiellement à capturer des paquets Ethernet circulants dans le réseau LAN (Cela est réalisé en masquant l'interaction

avec l'adaptateur du réseau Ethernet), et les transférer au programme de capture appelant. Etant donné que le système est lourdement dépendant, cela revient à l'implémentation, puisque celle ci est complètement différente dans les divers systèmes d'exploitation de la plate forme de Win32.
La section de capture des paquets de bas niveau du noyau de l'architecture, doit être rapide et efficace parce qu'il faut qu'elle soit capable de capturer aussi des paquets dans des réseaux LAN-Ethernet performants, c'est-à-dire des LANs possédants un Haut débit, des câbles de connexion puissants, un trafic important circulant dans le réseau et une perte de paquets limitée et restreinte, et cela en utilisant un pourcentage réduit des ressources systèmes. La section de capture doit aussi être générale et flexible pour être utilisée par les différents types d'applications (tels que : les analyseurs, les moniteurs réseaux, les applications de tests de réseaux, etc...). L'application de capture du niveau utilisateur reçoit les paquets Ethernet du système, les interprète et les traite, puis les affiches en claire à l'utilisateur d'une manière compréhensible et productive. Cette application doit être facile à utiliser, indépendante du hardware du système, modulaire et développable, afin qu'elle puisse supporter le grand nombre de protocoles réseaux et devrait aussi permettre d'augmenter le nombre de protocoles décodés d'une manière simple. Ces caractéristiques s'avèrent si importantes et essentielles à cause, du nombre de protocoles réseaux disponible actuellement et qui sont utilisés et manipulés par le réseau local Ethernet (tels que : IPX/ SPX, TCP/IP, NetBEUI, NetBIOS...) et de la vitesse dans laquelle ils changent et ils progressent. La figure suivante montre la structure de la capture avec une application de capture dans un même adaptateur réseau.

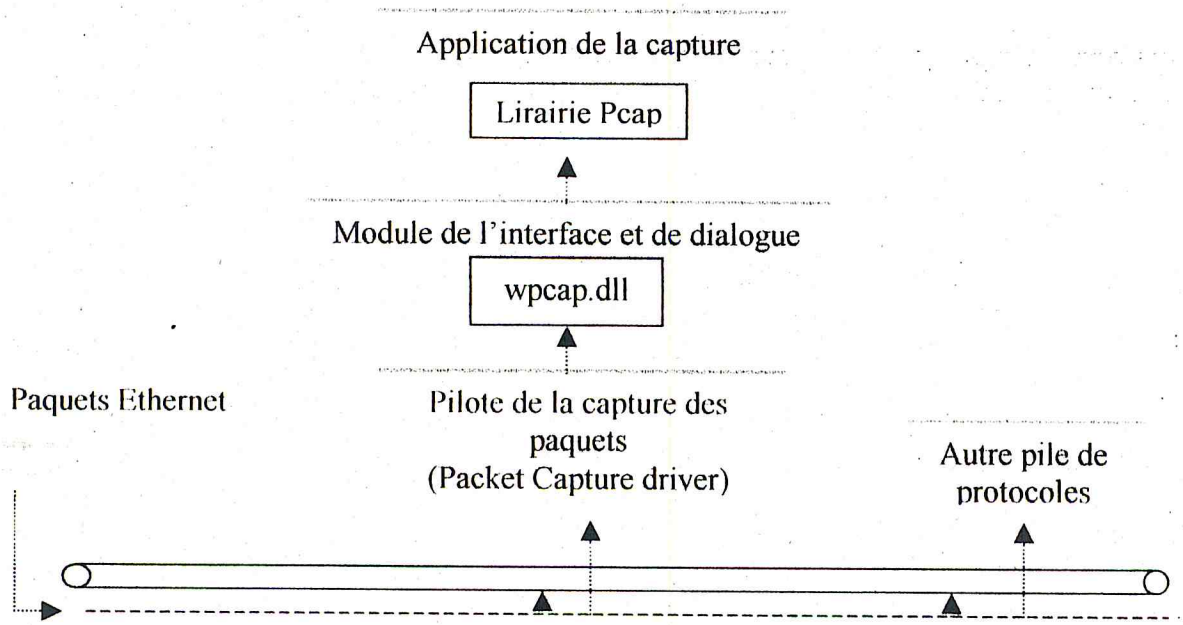


Figure 4 : Structure de la capture des paquets Ethernet

6- Structure interne de l'architecture de capture

L'architecture de capture définie au préalable, ajoute aux systèmes d'exploitation de la famille de Win32 la possibilité de capturer des données via un réseau, en utilisant l'adaptateur réseau installé dans la machine connectée à ce réseau. En plus, elle fournit à l'application de capture une interface de programmation d'application de haut niveau (appelé API, pour Application Programming Interface) qui simplifie la communication et le dialogue avec le niveau noyau, c'est-à-dire le plus bas niveau de l'architecture de capture. Cette architecture de capture subdivise le dialogue avec la carte réseau (appelée aussi adaptateur réseau) en trois éléments implémentés séparément et qui se présentent comme suit :

- Le pilote de capture des paquets Ethernet ;
- La librairie dynamique de bas niveau ;
- La librairie statique de haut niveau.

Cette architecture peut être utilisée pour créer de nouveaux outils de capture pour Windows. Dans cette section nous décrierons, la structure interne de cette dernière et le principe de base de cette architecture, et nous expliquerons quelques concepts qui seront utiles pour la compréhension du fonctionnement de cette architecture de capture. La structure de base interne est montrée schématiquement par la figure 5 :

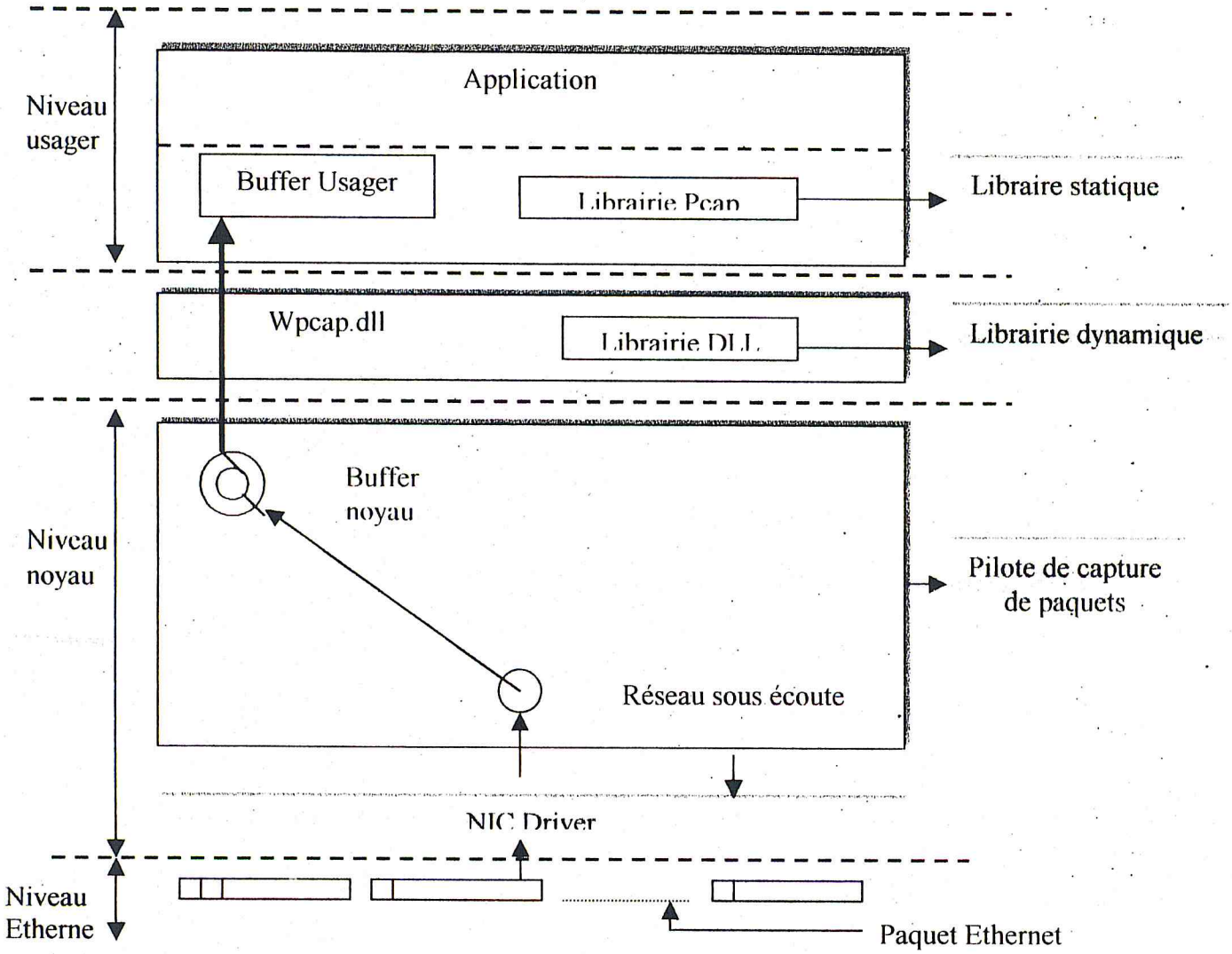


Figure 5 : Structure interne de l'architecture de capture

- Les flèches montantes vers le sommet de l'architecture indiquent le flux de paquets de données provenant du réseau vers l'application de capture de l'utilisateur.
- Les flèches orientées vers le bas de l'architecture indiquent l'acheminement des paquets de l'application vers le réseau, cela signifie que l'administrateur réseau a la possibilité d'interroger son réseau local en envoyant des requêtes via le réseau.
- La grande flèche entre le buffer du niveau noyau de l'architecture et celui de l'application de l'utilisateur indique que plus d'un paquet peut transiter entre ces deux entités en une seule lecture.

7- Principe de fonctionnement de la procédure de capture

Au niveau le plus bas, on retrouve l'adaptateur réseau, cet élément est utilisé pour lire les paquets Ethernet qui circulent dans le réseau.
Durant la capture, l'adaptateur réseau travaille souvent sous un mode de dialogue appelé « Mode Promiscous », ce mode de capture oblige l'adaptateur à accepter tous les types de paquets qui peuvent circuler dans le réseau. Le pilote de la capture (Appelé en anglais : BPF Packet Capture Driver) est le module software de plus bas niveau de la pile de capture, ce module dialogue et communique avec l'adaptateur du réseau sur lequel nous avons lancé le programme de capture pour obtenir les paquets circulant dans le réseau Ethernet. Ce pilote fournit à l'application une série de fonctions utilisées pour lire et écrire des données via le réseau, et cela se passe exactement au niveau de la couche « Liaison de données ».

Pour le module wpcap.dll ; celui ci travaille avec le niveau usager, mais il est séparé du programme de l'application de la capture utilisateur appelante. Wpcap.dll représente en fait une librairie de lien dynamique (DLL, pour Dynamic Link Library). Cette librairie sépare les programmes de capture du pilote de capture, fournissant ainsi une interface de capture indépendante du système, elle permet à ces programmes d'être exécutés sur différentes versions de win32 (Windows 95/98, Windows NT et Windows 2000), sans qu'elle soit recompilée à nouveau.

La librairie Libpcap (pour Librairy packet caplure) est une librairie statique qui est utilisée par la partie réservée à l'application de l'utilisateur, qui réalise la capture des paquets Ethernet. Cette librairie de fonctions utilise-les services exportés par la librairie de lien dynamique wpcap.dll et fournit au programme de capture du niveau utilisateur une interface de programmation plus souple et plus performante utilisant un langage évolué. L'interface utilisateur est la partie la plus élevée de la structure de la capture, elle gère l'interaction avec l'utilisateur (administrateur réseau) et affiche les résultats de la capture.

8- Niveau noyau

Le principal rôle de la partie noyau de l'architecture est de prendre les paquets circulant dans le réseau local Ethernet du niveau liaison de données et de les transférer au niveau application sans aucune modification.
L'application peut avoir accès à ce pilote de capture en utilisant des primitives de lecture, et d'autre d'écriture, et peut considérer l'adaptateur réseau de manière à ce qu'il soit similaire à un fichier normal, lisant et écrivant les données arrivant du réseau. Le pilote de capture peut aussi être utilisé par n'importe qu'elle application de capture développée sous la plate forme de Win32 qui à besoin de capturer et lire les paquets circulant dans le réseau pour d'éventuels traitements.
L'interaction avec le pilote de la capture passe toujours à travers la librairie de lien dynamique (wpcap.dll), cette dernière qui implémente un ensemble de fonctions permettant de simplifier la communication avec le pilote, évitant ainsi l'utilisation des appels systèmes ou les IOCTL (Input Output Control). Ce pilote dialogue avec le mécanisme de pilote de l'adaptateur réseau appelé aussi « Gestionnaire de la carte réseau » à travers le NDIS (Network Driver Interfacez Specification).
NDIS est une spécification par Microsoft d'une interface logicielle universelle d'accès aux cartes réseaux installées dans les PC, cette interface permet, au logiciel situé au niveau de la couche 2 du modèle OSI, d'utiliser n'importe qu'elle carte réseau, et est responsable de la gestion des adaptateurs réseau ainsi que la communication entre ces adaptateurs et les portions software qui implémentent les protocoles réseaux.

9- Module d'interface et de dialogue

Ce module constitue le niveau 2 de l'architecture de la capture. En fait, ce n'est rien d'autres qu'une API (Application de programmation d'application) qui peut être pour accéder aux fonctions du pilote BPF.

9-1- Définition

Le module le plus important qui permet de faire l'interface et le dialogue du pilote de capture des paquets avec l'application de capture du niveau utilisateur, représente en fait une librairie dynamique de fonctions appelée wpacp.dll, celle-ci implémente une série de fonctions qui réalisent une communication plus simple avec le driver. Cela évite l'utilisation des appels systèmes ou IOCTL dans les programmes usagers. De plus, cette DLL fournie des fonctions pour manipuler l'adaptateur réseau, la lecture et l'écriture des paquets sur le réseau....
Ainsi, elle facilite l'écriture des applications de capture indépendamment du hardware du système. Par l'utilisation de cette API, la même application de capture s'exécutera sous les différentes versions de Windows 95/98,NT et 2000 sans aucune modification. Cette caractéristique permet d'écrire une seule version de la librairie de fonction Libpcap pour la famille du système d'exploitation windows.

9-2- Packet Driver (wpcap.DLL) et la librairie de capture (libpcap)

Pour écrire une application qui réalise la capture et la lecture des paquets dans le réseau, nous avons besoin d'utiliser des primitives et des fonctions spécifiques pour pouvoir dialoguer avec le pilote de la capture, qui lui-même va dialoguer avec le gestionnaire de la carte réseau Ethernet, pour enfin arriver au niveau de l'adaptateur (interface réseau), et faire remonter les informations et les messages échangés entre les stations qui sont connectées au même réseau.
Deux librairies de fonctions permettent de dialoguer avec l'adaptateur réseau pour faire remonter ces paquets de données :

- La librairie de lien dynamique (wpcap.DLL) ;
- Packet Librairie ou la librairie de capture (Libpcap).

La librairie Libpcap utilise aussi bien les fonctions de la librairie dynamique wpcap.dll, mais elle fournit un environnement de programmation plus performant, immédiat et facile à utiliser. Avec Libpcap, nos opérations telles que, la capture des paquets de données, la création des filtres de capture, ou la sauvegarde de la capture dans un fichier sont implémentées en sécurité et d'une manière plus souple à utiliser. Libpcap est capable d'offrir toutes les fonctions sollicitées par le réseau pour faire une capture.

10- Module de capture utilisateur

C'est le module le plus important, c'est le programme de capture et d'analyse des trames de l'utilisateur.

11- Exemple d'une trame capturée

La figure suivante montre une trame capturée par notre logiciel, cette capture affichera les champs constituant la trame Ethernet en hexadécimale:

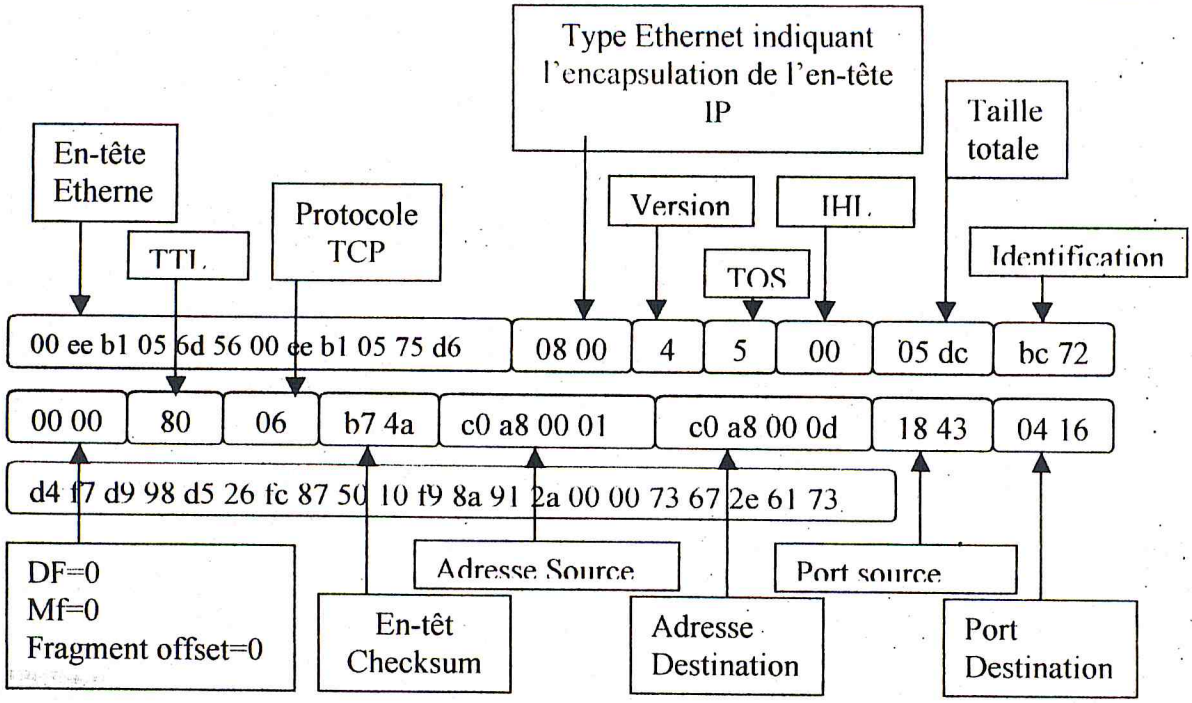


Figure 6 : Exemple de trame capturée

IV- Le ping (La détection des stations du réseau)

1- Scruter l'état des stations

Dans cette partie du projet, nous allons tester toutes les stations du réseau, c'est à dire voir si elles sont active ou pas (cela se fait par le ping)

2- Définition du ping

Le ping est une commande qui permet de mesurer le temps de latence d'un site. C'est à dire le temps que met une machine pour répondre à une sollicitation réseau.
Il permet de savoir à tout moment quels sont les ordinateurs connectés sur notre réseau, et d'être avertit en cas de déconnexion d'ordinateur. On peut l'utiliser avec une adresse IP ou un nom de machine.

La commande ping a principalement pour but de tester l'accessibilité d'un node. La machine qui envoie un ping vers une autre machine, attend un écho de son appel pour s'assurer que celle-ci est bien disponible.
Le message de ping doit suivre le routage normal de IP à travers les passerelles et routeurs ... Il utilise pour cela le protocole ICMP encapsulé dans la trame IP.
Le champ "CODE" du message ICMP peut donner des informations sur le résultat du test.
Le retour d'un PING (echo reply = pong) donne généralement le temps mis par le message pour faire l'aller-retour (RTT = round trip time) jusqu'au destinataire.

3- Principe de fonctionnement

La figure suivante montre le déroulement du programme :

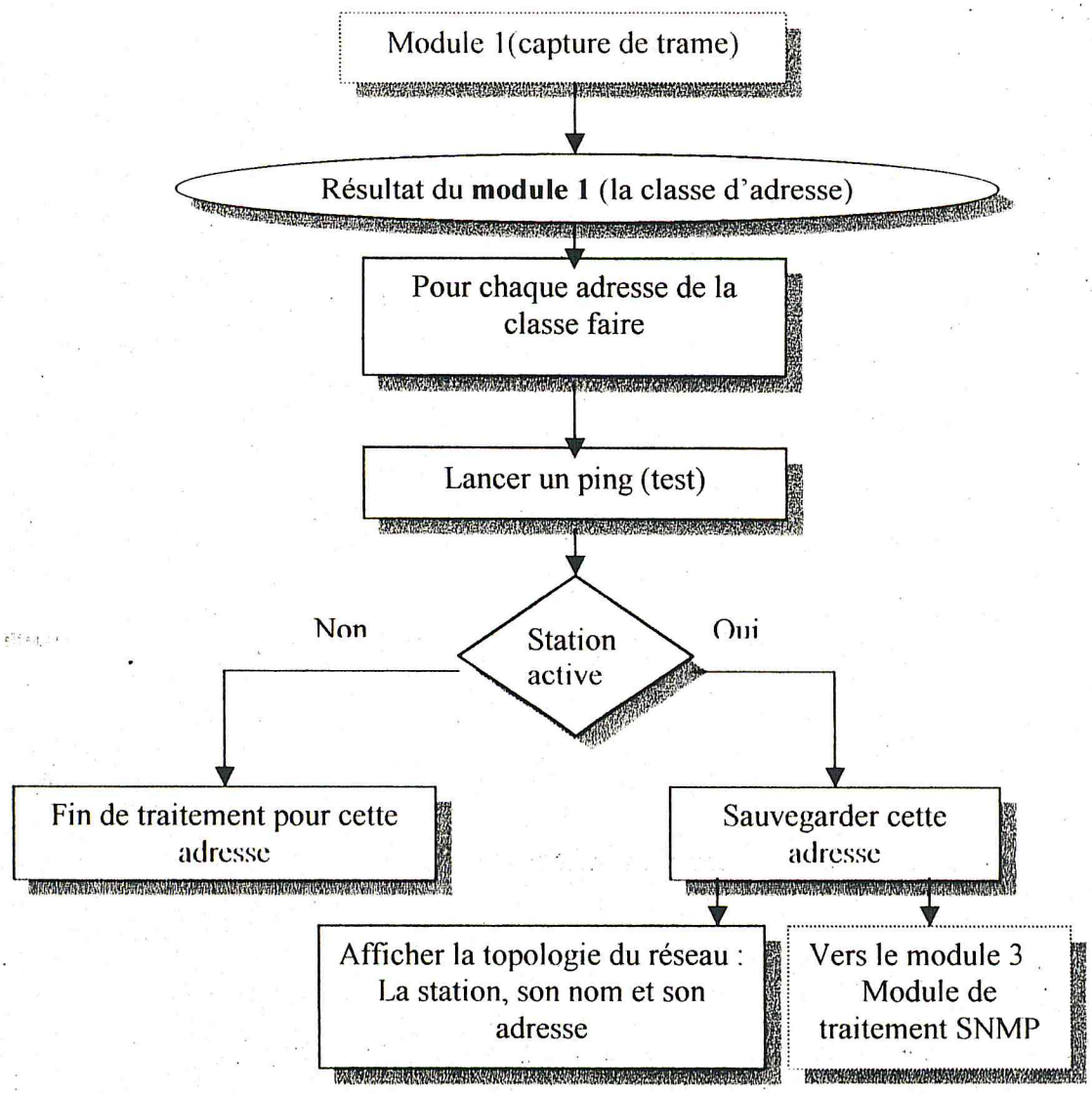


Figure 7 : Algorithme de détection des stations actives dans le réseau

Cette partie nous permet de savoir si une machine est connectée au moment où on lance le programme avec la machine où l'on travaille. Pour ce faire, elle envoie, à toute adresse de la plage d'adresse spécifiée à l'avance par le module1 (capture de trame), un ping. Si elle est connecté alors on sauvegarde l'adresse pour des traitements ultérieurs du module 3 (module de traitement SNMP).

V- Traitement de la cartographie (SNMP) et détection des alertes

1- Introduction

SNMP (Simple Network Management protocol) est un protocole utilisé dans le domaine de la supervision des réseaux.

SNMP décrit le langage que les agents et les stations de gestion utilisent pour communiquer ; c'est un protocole de type question/réponse asynchrone.
Il harmonise les méthodes de surveillance et de contrôle et offre ainsi un moyen de gérer convenablement des réseaux hétérogènes.

SNMP permet à deux entités distantes (un agent et un gestionnaire de réseau) de communiquer des informations.

Le protocole SNMP permet la remontée d'informations des équipements vers le gestionnaire, afin d'identifier les éléments raccordés au réseau et de connaître leurs états.

L'agent SNMP est disponible sur de nombreux équipements : dans les systèmes d'exploitation (sous forme de service) mais aussi dans les équipements réseaux (routeurs, commutateurs, imprimantes, etc.) comme entité logicielle.

SNMP utilise le protocole UDP (User Datagramm Protocol) pour transporter les données entre les agents et le manager. Il travaille donc en mode non connecté.

SNMP est un moyen simple et facile pour établir la cartographie du réseau (tous ce qui concerne les requêtes de demande, de modification et de réponse) ainsi, c'est un moyen indispensable pour gérer les dysfonctionnements et les alertes en cas d'erreurs et de problèmes.

2- Description

Il existe deux stratégies de collecte d'information de gestion entre le gestionnaire et l'agent. La première est basée sur un mécanisme de scrutation (polling). Elle consiste pour le gestionnaire à demander à l'ensemble des agents du réseau, des informations sur l'état de leurs ressources (requêtes GET, GET_NEXT ou SET). La deuxième méthode est basée sur un mécanisme d'événements : l'agent prend l'initiative d'avertir le gestionnaire des alertes en envoyant des messages de façon asynchrone (TRAP).

Chaque agent SNMP répond ainsi à des interrogations en provenance de la station de surveillance et peut également envoyer des messages d'alarme limités à cette station.

3- Le modèle de gestion de réseau SNMP

Nous avons utilisé dans cette partie un système de gestion de réseau standard via le protocole SNMP. Notre système de gestion se compose de:

- Nœuds administrés (Manager Node : MN) appelés aussi éléments de réseau : ils configurent les différentes ressources tels que les routeurs, les hubs, ainsi que tout dispositif qui compose le réseau.
- Un ou plusieurs agent(s): est l'entité logicielle qui s'exécute dans les éléments du réseau. L'agent a l'accès direct aux parties matériels et logicielles des équipements. Il est interrogé à distance par les stations de gestion et fournit les informations (exécute des instructions) et aussi il alerte le manager des disfonctionnements.
- A l'autre extrémité se trouve la ou les plate-forme(s) d'administration, c'est le manager (NETWOEK MANAGER STATION : NMS). Ce logiciel est souvent graphique, il réside dans une station d'administration de réseaux ; il interroge, via le protocole d'administration SNMP, les agents et présente les résultats à l'administration.
- Le protocole SNMP : le rôle de ce protocole est de véhiculer les informations de gestion entre les managers et les nœuds gérés.
- La MIB : (Management Information Base) elle décrit les informations de gestion d'un nœud géré, mais elle ne donne cependant pas leurs valeurs. C'est à l'agent donc de récupérer l'information à chaque interrogation de la MIB.

4- La configuration de notre système

La figure suivante montre les principales opérations échangées entre l'agent et la NMS :

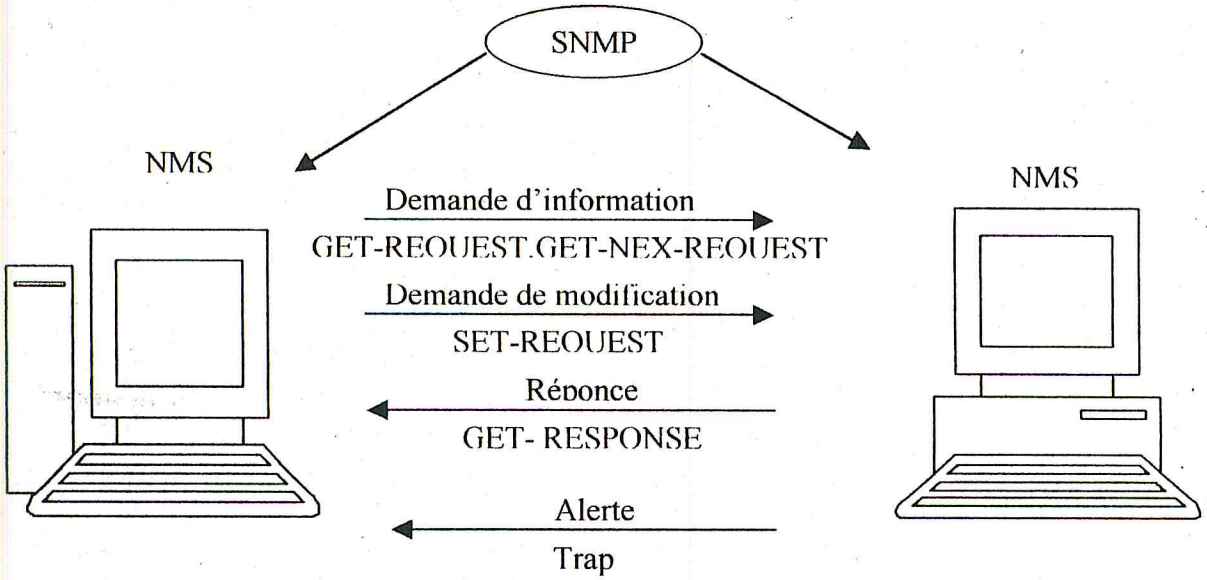


Figure 8 : Principe de fonctionnement

5- Le modèle fonctionnel de notre système

SNMP fonctionne avec des requêtes, des réponses et des alertes.
Notre modèle fonctionnel se divise en deux parties :
La première partie : Les stations de gestion **NMS** envoient une série de requêtes (interrogations) à l'agent, sur chaque élément de réseau afin d'observer son fonctionnement et de requérir des informations sur son état (nom de l'ordinateur, son système d'exploitation, le nom du routeur si c'est un routeur et le nombre de trames qui le traverses ...).
Les managers effectuent une interrogation périodique des agents de manière à vérifier leurs états.
Dans la deuxième partie : Les agents lancent des alertes asynchrones lorsqu'ils veulent avertir le NMS d'un problème tels que des erreurs de transmission ou des disfonctionnements dans le logiciel ou dans le matérielle. Mais ces alertes spontanées sont limitées.
SNMP est un service qui fonctionne au-dessus du protocole de transport UDP, il a donc besoin d'un numéro de port pour communiquer. Deux ports lui sont réservés : 161 et 162.
La station d'administration émet par le port 161 ou autre selon l'application utilisée en direction de l'agent qui reçoit par le port 161.
L'agent répond par le port 161 à la station d'administration qui reçoit cette réponse aussi par le port 161.
L'agent émet le message d'alarme par le port 161 mais la station de gestion le reçoit par le port 162.

6- Organigramme décrivant l'algorithme (partie SNMP)

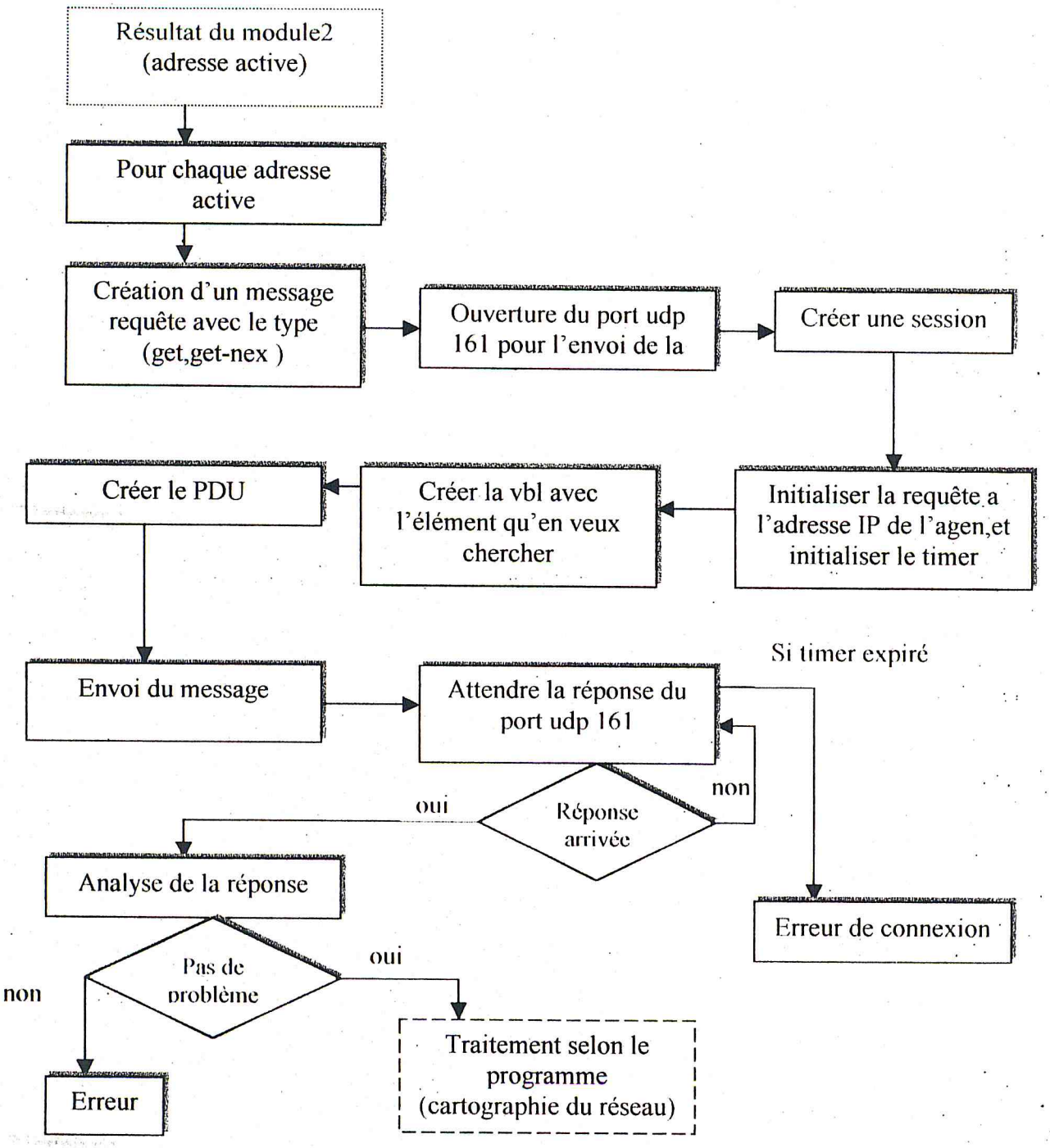


Figure 9 : Algorithme SNMP

7- Le principe de fonctionnement

7-1- Première partie

Notre application permet de récolter et d'analyser les données relatives aux équipements logiciels et matériels connectés au réseau. Pour cela, elle doit effectuer une série de questions (Console Agent) pour pouvoir arriver à ces fins.

Processus d'envoie de message (par le manager):

L'application questionne un agent en lui envoyant une requête de demande PDU get-request. Pour mieux comprendre ce processus nous avons choisi un exemple sur la récupération du nom de l'ordinateur :

Pour obtenir le nom d'un ordinateur distant, on lui envoie une requête de demande (PDU) qui a la structure suivante :

Structure de la PDU GET, GET-NEXT, SET, Response :

Type de PDU	ID demande	Etat d'erreur	Index d'erreur	Var Bind Liste

Figure 10 : Structure de la PDU GET

Type de PDU : 0 : GetRequest ; 1 :GetNextRequest ; 2 :GetResponse ; 3 : SetRequest
ID demande : contient un entier, il fait correspondre une réponse à une demande particulière.
Etat d'erreur : utilisé uniquement par l'agent dans l'envoi de la PDU GetRespons à la NMS. Il contient une valeur qui varie entre 0 et 5 :
Index d'erreur : utilisé uniquement dans la PDU GetRespons afin de fournir plus d'informations sur l'erreur détectée par l'agent.
Var bind list : (variable binding liste c'est à dire liste de liaison de variable).

Structure de la PDU GET envoyée :

0	ID demande	0	0	1.3.6.1.2.1.1.5.0

Figure 11 : Exemple de la PDU GET envoyée

Le champ Type de PDU =0 parce que c'est le numéro de la requête Get-Request
Le champ ID demande c'est l'identifiant de la demande (elle se remplie selon le programme)
Les champs Etat d'erreur et Index d'erreur valent zéro parce que c'est une demande (voir chapitre III VII-1)
Le champ Var Bind Liste c'est la liste de toutes les variables, il contient les objets sur lesquelles s'opèrent la requête, dans notre cas (le nom de l'ordinateur) est 1.3.6.1.2.1.1.5.0 pour sysName.

Cette PDU est envoyée avec la version de SNMP utilisée et le nom de la communauté ;

Version	Communauté	PDU

Figure 12 : Le paquet SNMP envoyé

La version : un entier indiquant la version utilisée par le logiciel SNMP, zéro pour la version1.
La communauté décrite par une chaîne de caractère, permet de créer des domaines d'administrations. La communauté est Public par défaut.
PDU est constituée d'une demande du manager ou d'une réponse d'un agent.

Processus de recherche de réponse (exécuté par l'agent) :
A l'autre extrémité l'agent SNMP reçoit la demande (PDU envoyé par l'application) et procède comme suit :

1. Il compare la version de SNMP reçut pour voir si elle est compatible avec sa version ou pas.
2. Il fait correspondre le nom de la communauté reçu avec celui de son ordinateur pour authentifier l'expéditeur.
3. Il récupère le Type de PDU pour savoir si c'est une requête get, getnext ou set pour savoir comment réagir.

4. Il récupère l'ID demande reçu pour référencer sa réponse avec le même ID demande.
5. Il récupère le chemin de l'élément cherché par l'application dans le champ Var bind List pour qu'il puisse arriver à sa valeur.
6. Il récupère la valeur de l'élément cherché.

Processus d'envoi de réponse (par l'agent):
L'agent envoie à l'application sa réponse sous forme de GET-response.
La PDU GET-response garde la même structure que la PDU GET-request.
L'agent envoi la réponse dans le champ Var bind List, type de PDU = 2 (pour GET-response), dans le champ approprié, le champ ID demande garde le même ID demande que la PDU de l'application émettrice.
Les champs Etat d'erreur et Index d'erreur valent zéro s'il n'y a pas d'erreur. Dans le cas contraire, ce champ contient un identifiant d'erreur (voir chapitre III VII-1)
Exemple de PDU de réponse :

2	ID demande	0	0	ObjectID, Valeur

Figure 13 : Exemple de la PDU Response

Cette PDU est envoyée avec la version de SNMP utilisée et le nom de la communauté, comme pour la PDU GET-request.

La figure suivante montre le parcourt du message SNMP dans l'agent :

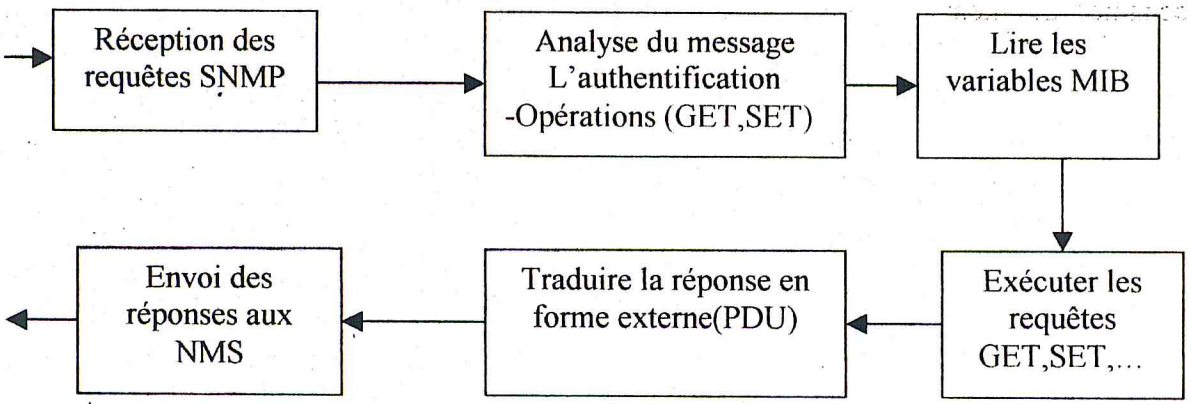


Figure 14 : flux d'un message SNMP à travers un agent

Processus de réception de réponse par la NMS :
Le processus de réception de réponse par la NMS est le même que celui de la réception d'une demande par l'agent, à part que le champ Var bind List de la PDU ne contient pas la variable à chercher, mais plutôt la valeur de la réponse.
La figure suivante montre le parcours d'une réponse dans la NMS.

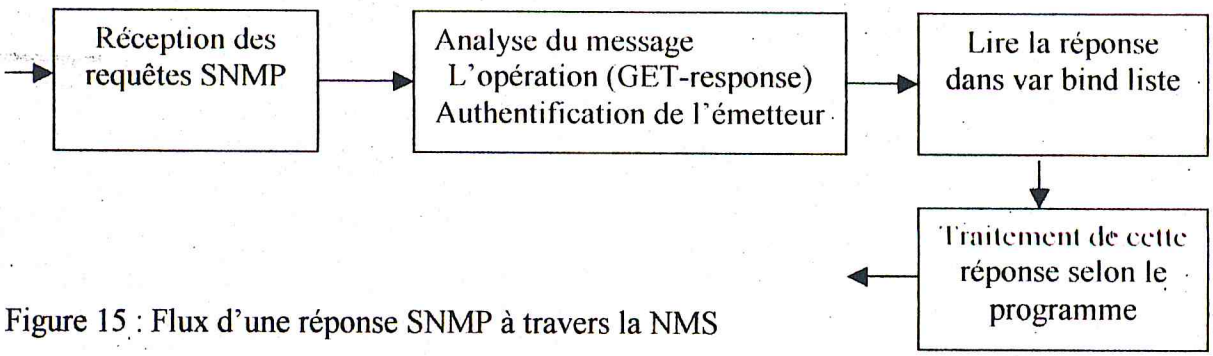


Figure 15 : Flux d'une réponse SNMP à travers la NMS

7-2- Deuxième partie

Cette partie concerne les alertes détectées par l'agent, c'est les disfonctionnements signalés dans l'équipement. Lorsqu'il y a un disfonctionnement l'agent envoi une requête (Trap) à la NMS où l'application réside, (l'adresse de la NMS est enregistrée lors de la configuration de l'agent pour que celui ci puisse lui envoyer des alertes).
La structure du PDU trap est la suivante.

Type de PDU	Entreprise	Agent	Interrupt générique	Interrupt spécifique	Horodatage	VarBindList

Figure 16 : Structure de la PDU TRAP

Type de PDU : toujours égal à 4.
Entreprise : permet de spécifier le type d'objet qui a généré l'interruption.
Adresse agent : contient l'adresse IP de l'agent.
Interruption générique : un entier représentant l'une des valeurs de trap prédéfinies en standards pour le protocole SNMP. Ces valeurs ou traps génériques sont au nombre de 7.
Interruption spécifique : Si le champ interruption générique vaut 6 (spécifique entreprise), alors le message d'interruption est spécifique à cette communauté de gestion de réseau.
Champ horodatage : contient l'heure à laquelle le trap a été génère, en centième de seconde.
VarBindList : se compose d'une liste de variables et de valeurs associées à l'interruption.
Exemple de la PDU Trap :

4	x.x.x.x.x.x.x	y.y.y.y.y.y	5	Interrupt spécifique	zzzzzz	j.j.j.j.j.j.j.j

Figure 17 : Exemple de PDU TRAP

La figure suivante montre le parcours d'une PDU TRAP dans la NMS.

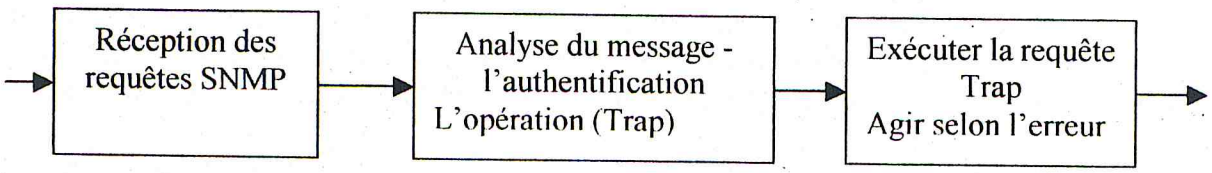


Figure 18 : Flux d'un TRAP SNMP à travers la NMS

VI- La qualité de service

1- Introduction

En quittant le monde académique et militaire pour se développer dans la société, les contraintes posées sur le service de transfert offert par l'Internet ont changé. À ses débuts, Internet avait pour seul objectif de transmettre les paquets à leur destination. Conçu pour le transport asynchrone des données, IP (Internet Protocol) n'a pas été prévu pour les applications en temps réel comme la téléphonie ou la vidéo, très contraignantes. Le besoin en équipements de plus en plus fiables, d'un bout à l'autre du réseau, est donc devenu incontournable. Cependant, les défauts rencontrés sur les réseaux (perte de paquets, congestion,....) ne peuvent pas être surmontés sans une rénovation profonde de l'architecture. En effet, l'Internet commercial s'accommode mal du service "au mieux" qui a prévalu depuis ses débuts. Les utilisateurs souhaitent avoir un service qui corresponde à leurs besoins. Les fournisseurs d'accès Internet souhaitent allouer efficacement les ressources afin d'accroître leur activité au prix d'un investissement minimum.

Une partie élastique, elle correspond à un débit de paquets opportunistes. Aucune garantie n'est apportée à ces paquets. Ils sont acheminés par le réseau sur le principe du "au mieux". En cas de congestion, ce sont ces paquets qui seront éliminés en premier. Le débit opportuniste doit varier en fonction de l'état des ressources utilisées d'où son caractère élastique. Il demande à l'utilisateur du service réseau d'adapter son débit opportuniste à la capacité du moment du réseau.

- Enfin, un service à garanties fermes (Garanteed Service : GS) pour des applications rares qui posent des contraintes de QoS fortes et qui de plus ne supportent pas des variations de la QoS. Ce service est destiné aux flots déterministes. Le service est conçu pour des flots continus ou réguliers. Le service GS présente des propriétés très intéressantes dans un contexte multicast. Le niveau de pertes très faibles de paquets de ce service le rend très avantageux d'utilisation pour un service multicast fiable.

4- QoS : Priority Queuing en standard sur les routeurs Perle

Les fonctions de gestion de la qualité de service (QoS : Quality Of Service), popularisées par leur activation par défaut dans Microsoft Windows/XP, prennent de l'importance sur des réseaux IP toujours plus chargés et des applications toujours plus complexes. Un routeur doté de fonctions de QoS permet de garantir un niveau de performance prévisible même en présence de saturations sur le trafic réseau général.

La fonction QoS du Priority Queuing a été incluse dans les caractéristiques de base des routeurs Perle P800, P1730 et P2600. La mise en file d'attente par priorité (priority output queuing) permet à l'administrateur de réseau de définir quatre (4) niveaux de priorités pour les flux réseau : haute, normale, moyenne ou basse, pour chaque interface et chaque protocole. Tout paquet entrant dans le routeur est affecté à l'une de ces quatre files d'attente de sortie. Les paquets de la file de plus haute priorité, sont émis les premiers, puis ceux de la deuxième file lorsque la première est vide, et ainsi de suite. Ce mécanisme permet de s'assurer que durant les périodes de congestion de la bande passante, les données de priorité haute ne sont pas retardées par les flux de plus faible priorité.
Cette fonction, habituellement présente sur les grands routeurs, est regrettablement absente des services offerts par les routeurs d'entrées de gamme, d'autres marques sont souvent mises en concurrence avec le P800. La mise à jour vers une version de logiciel de routage incluant cette nouvelle fonction est gratuite pour tous les possesseurs de routeurs Perle.

5- Les mécanismes requis par la qualité de service

• Contrôle d'admission : acceptation/rejection d'une demande d'appel à chaque étape au long du chemin
• Fonction de police : garantir que l'émetteur n'émet plus que promis afin de garder les garantis pour les autres
• Gestion de queues : fournir le service promis

6- Les approches existantes de la qualité de service

Deux approches de qualité de service de l'IETF existent : IntServ et DiffServ.

6-1- Caractéristiques de l'IntServ (Integrated Service)

• héritage : travaux débutés en 1994
• réservation par flot

• problèmes de résistance au facteur d'échelle
• complexe à gérer sur le plan utilisateur

6-2- Caractéristiques de DiffServ (Differentiated Service)

• pour faire face aux défaillances d'IntServ (travaux débutés en 1998)
• notion de classes de services
• traitement sur une agrégation de flots
• complexe à gérer au niveau du dimensionnement et de la gestion des ressources
• implémenté dans les routeurs

7- IntServ (Integrated Service)

Son principe est orienté délais, il a trois classes de service :
• Guaranteed Service (le service garanti)
• Controlled Load (la charge contrôlée)
• Best Effort (moindre effort)

1. Guaranteed Service

Motivation : Limite les délais pour les applications inflexibles
Le réseau garantit :
• un débit
• un délai maximal de transfert
Le réseau ne garantit pas :
• la gigue
• le taux de pertes

2. Controlled Load

Motivation : Les applications adaptatives marchent assez bien dans un réseau non congestionné
Pas de garantis quantifiables
Le réseau garantit : le trafic ne va pas subir des congestions

3. Best Effort

Motivation : Fournir le service déjà existant dans l'Internet
Moindre effort classique.
Pas de garantis pour :
• les délais
• les taux de pertes
• etc.

7-1- Composants de Intserv

Architecture d'un Routeur IntServest présentée dans la figure suivante : **[Tim 03]**

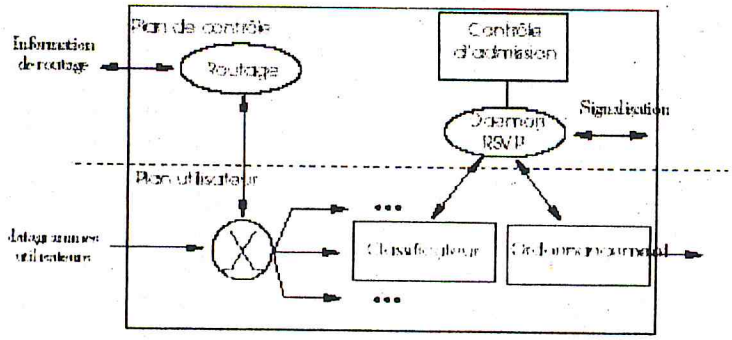


Figure 19 : Architecture d'un routeur InterServ

1. Contrôle d'admission

Attention: différent du contrôle d'accès (policing)

Le routeur :

- Décide si un nouveau flot peut être supporté
- La réponse dépend :
  - De la description du flot
  - De la classe de service demandée

2. Classification

Le « packet classifier » du routeur :

- Oriente les datagrammes selon la QoS demandée
  - Création de différentes files d'attente
- Associe chaque paquet avec la réservation appropriée

3. Ordonnancement des émissions

Le « packet scheduler » du routeur :

- Ordonnance des datagrammes afin de respecter la QoS demandée

Ordonnancement Problèmes avec le « FIFO »

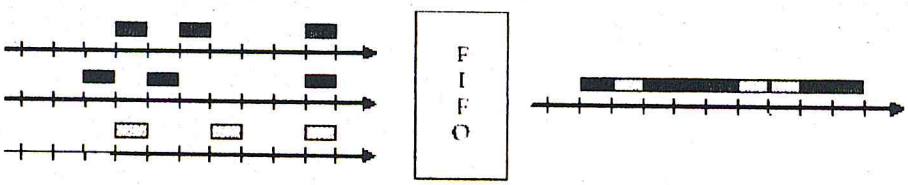


Figure 20 : Ordonnancement (problèmes avec le FIFO)

- Modifications de l'espacement temporel
  - introduction de la gigue
- Impossible de donner des priorités aux flots

Nouvel ordonnancement Le WFQ (« weighted fair queuing »)

- discipline de service équitable et pondéré
- permet de garantir le débit
- une approximation d'équité « max-min » pondérée

Équité « max-min »

Principes :

- Satisfaire les demandes les plus petites en premier
- Satisfaire les autres dans une manière égale

WFQ pour les garantis
Ressource : nombre de cycles dans la fonction d'acheminement
Flots prioritaires :
• poids supérieurs, alors proportionnellement plus de cycles
• temps de service plus court
Autres flots :
• poids inférieurs
Problème avec le WFQ
• difficile à implémenter

4. Protocole de signalisation
RSVP (Ressource Reservation Protocol): un protocole pour la réservation de ressources
• Allocation de ressources nécessaires pour fournir le service

RSVP (Ressource Reservation Protocol :
C'est un protocole de signalisation, il Informe les besoins applicatifs aux réseaux. Conçu pour demander des réservations IP, vu comme le protocole de signalisation de IntServ
Principe de RSVP:
La signalisation est constituée d'un flux de messages SDWK et UHVY
Pas de réservation pour ce flux
Remise sans garantie et non acquittée

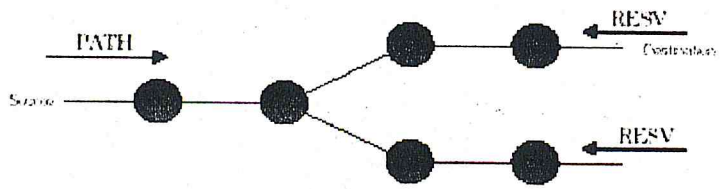


Figure 21 : Principe de RSVP

Chaque routeur RSVP traversé par un flux RSVP mémorise un état de ce flux :
Soft state

- Notion de contexte applicatif
- Rafraîchi par les messages UHVY
- Après un certain délai L de non-réception, l'état est détruit

Libération immédiate de la réservation

- Messages teardown (démolition) **[Tim 03]**

Messages PATH
Émis « périodiquement » par la source interceptée par les routeurs, suit le même chemin que les données.

Messages RESV
Emis par le(s) destinataire(s)
Demande une réservation en bande passante
Prend le chemin inverse des messages path, en multicast, les messages resv sont fusionnés

Caractéristiques de RSVP

- Protocole de bout en bout
- Utilise IP.
- Réservation simplex: notion de session définie par rapport à la « destination »
- Réservation pour des communications unicast ou multicast

- Orienté récepteur
- Récepteurs hétérogènes
- Styles de réservation différents pour les différentes applications
- Supporte le changement de route ou de membre du groupe

7-2- Avantages de IntServ

- Services proches des différents types d'application, ex: GS pour les applications critiques intolérantes
- Conçu pour fournir des garanties absolues
- Le flot peut être contrôlé par le routeur
- QoS pour l'unicast ou multicast
- Styles de réservations tendent à augmenter le taux d'utilisation des ressources réservées
- Adaptation « automatique » au changement de routes

7-3- Inconvénients de IntServ

- Service de bout en bout garanti si tous les routeurs sont Intserv
- Problème de facteur d'échelle
- Impraticable pour les flots à durée de vie courte
- Facturation du service complexe
- RSVP complexe

8.Philosophie DiffServ (Differentiated Services)

Conforme à la philosophie Internet
• simple au cœur
• complexe à la périphérie
• gestion distribuée par domaine

8-1- L'architecture de Diff-Serv

L'architecture s'applique au domaine d'administration, et reprend la séparation des fonctions entre la frontière et le cœur d'un domaine d'administration.
Un domaine se définit comme une portion contiguë de l'Internet contrôlée par une même autorité administrative. La frontière d'un domaine d'administration est marquée par un routeur de bordure. Ce routeur joue un rôle diffèrent de celui situé au cœur du domaine. La figure suivante montre le placement des différents routeurs dans le domaine d'administration. Cette architecture s'inscrit dans le même paradigme que l'Internet qui est : « de reléguer la complexité dans les extrémités du réseau et de laisser le cœur du réseau aussi simple que possible ». Cette architecture conduit à procéder à un simple ordonnancement des flots au cœur du réseau et au contrôle du trafic en bordure.

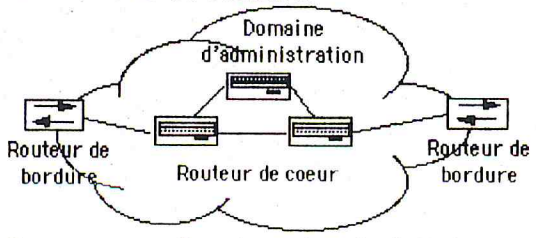


Figure 22 : Composants d'un domaine d'administration Diff-Serv

Le routeur de bordure a en charge la surveillance et le conditionnement du trafic entrant. Ces tâches sont complexes et mettent en jeu une grande variété de contextes. Elles servent à limiter la quantité de trafic injecté par chaque utilisateur dans le domaine. Elles sont essentielles et empêchent que les flots du service soient perturbés par la congestion. Les contrôles sur le trafic entrant s'appliquent au niveau du flot utilisateur. Le flot utilisateur peut être soit le trafic issu d'un site, soit celui généré par une application. Le résultat du conditionnement se traduit concrètement par un marquage des paquets admis dans le domaine, par la suppression des paquets excédentaires, ou par la remise en forme du flot (assurer un espacement temporel entre les paquets).

8-2- Structure fonctionnel d'un routeur

La structure fonctionnelle d'un routeur IP peut se représenter selon la figure suivante :

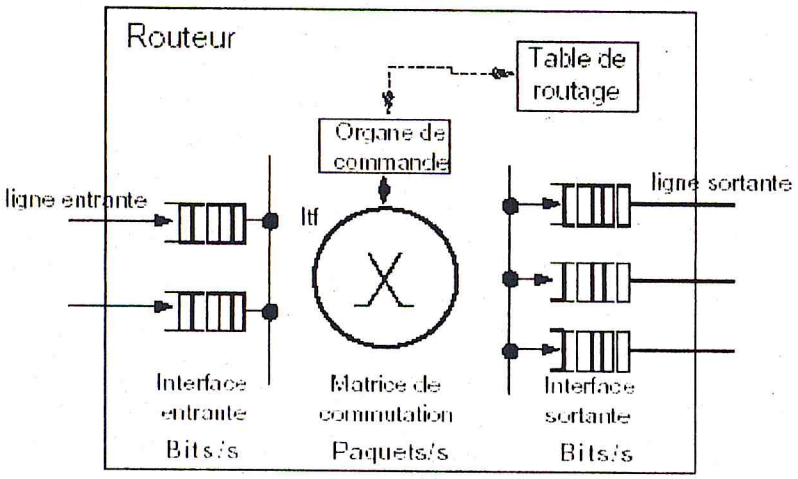


Figure 23:Structure d'un routeur.

Les fonctions de QoS de niveau IP se localisent dans les interfaces d'entrée et de sortie du routeur.
Les fonctions mises en oeuvre dépendent du rôle du routeur dans l'architecture (c'est à dire de bordure ou de cœur). Un routeur de cœur comportera uniquement les fonctions lieés au relayage; elles sont localisées dans l'interface de sortie. Ces fonctions visent à allouer efficacement les ressources que sont la bande passante et la mémoire de la file d'attente, en fonction du comportement de relayage propre à chaque classe de paquets. Un routeur bien dimensionné suppose que la capacité de commutation de la matrice est supérieure à la somme des débits entrants. Dans ces conditions, la file d'attente des interfaces d'entrée ne peut pas être saturée. La congestion se manifeste donc uniquement sur la file d'attente de l'interface de sortie. Il est donc du rôle des fonctions de relayage de protéger les flots à QoS de la congestion, afin de maintenir un niveau de service stable.
En plus des fonctions de relayage qui sont identiques aux fonctions d'un routeur de cœur, le routeur de bordure conditionne le trafic entrant dans le domaine. Ces dernières fonctions sont placées dans l'interface d'entrée. L'interface d'entrée en question est celle qui reçoit le trafic de l'extérieur du domaine. Cette interface peut se qualifier d'interface amont.

8-3- Principes de QoS (Conditionnement)

Actions décrites dans le contrat :

- Classifier
  - identification et classification des trafics
  - interface, ACL (port, TOS…)
- Marqueur (marker)
  - modifie le codepoint, donc le PHB pour donner un niveau de priorité différent (marquage ou remarquge des champs TOS ou DSCP)
- Testeur (meter)
  - test le niveau de conformité au profil
- Effaceur (dropper)
  - détruit le datagramme non conforme, modification des caractéristiques sémantiques du flot
- Remise en forme (shaper)
  - Modification des caractéristiques temporelles du flot

8-4- Avantages DiffServ

- **Traitement complexe en périphérie,**
  Concentration de trafics faibles
  Croissance du domaine, augmentation de la périphérique
- **Discrimination pour un réseau commercial**
  "Meilleur service pour ceux qui paient plus"
- **Pas de délai d'établissement ou de signalisation**
  Reste en mode non connecté
  Efficace pour les flots à durée de vie courte
- **Provisionnement du domaine**
  Méthode à la discrétion de l'administrateur
- **Classification**
  Simple
- **Marquage**
  Peut être effectué par le routeur de bordure
- **Découpage d'un domaine en plusieurs réseaux virtuels**
  Performance de chaque réseau par sa charge admissible

8-5- Inconvénients DiffServ

- **Service de bout en bout :** concaténation d'agréments et de politiques locales
- **Complexité dans :** le provisionnement du réseau et la configuration
- **Echelle de temps différente**
  Charge de trafic et le provisionnement
- **Difficile de garantir** l'absence de congestion locale malgré une charge connue
  Mauvaise répartition du trafic
  Changement de routes
  - Solution statique:
  Contrat de service entre paires de routeurs de bordure "Route pinning" mais perte de robustesse
  - Solution dynamique:
  Contrôle d'admission sur la route; modèle du "bandwidth broker"
  Très complexe
- **Difficile de garantir** la priorité pour des flots de classes différentes
  Signalisation des besoins quantitatifs de bout en bout

- **Orienté émetteur**

Signalisation du profil du récepteur

- **Multi-destination**

9- Gestion de la bande passante

D'un point de vue de gestion de la bande passante, les trois services se partagent la capacité selon le schéma de la figure suivante. Cette figure montre comment s'effectuerait le partage de la bande passante dans le temps. La bande passante laissée libre par le trafic GS est occupée par les trafics AS et BE proportionnellement au poids qui leur a été affecté par l'administration réseau. Le trafic du service BE utilise la bande passante laissée libre par le service AS.

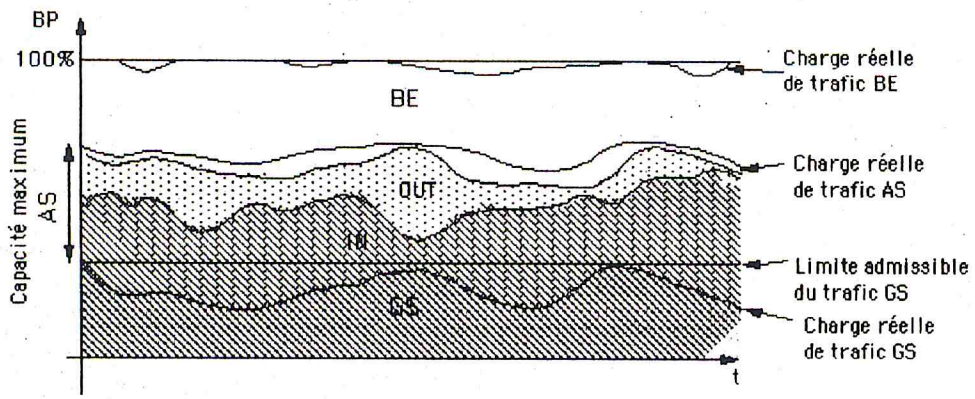


Figure 24 : Partage de la bande passante par les 3 services

Les expérimentations auront pour but, de vérifier la qualité des services offerts mais également si la bande passante est effectivement gérée de la manière présentée.

9-1- Alternatives : augmenter la bande passante

Pour résoudre les problèmes de qualité de service sur Internet, l'une des techniques consiste à augmenter fortement la bande passante (ce que l'on nomme overprovisionning, ou abondance de ressources). La méthode n'est, bien entendu, pas toujours la plus économique. De même, afin de commuter les paquets à des vitesses de l'ordre du terabit/s (un million de mégabit/s) ou de mettre en œuvre des algorithmes de routage sophistiqués, l'augmentation de la puissance de calcul, et donc de la puissance des processeurs, est une technique souvent utilisée. Enfin, la mise à disposition de terminaux intelligents capables de traiter des paquets de données ainsi que les nouvelles technologies d'accès (xDSL, câble, BLR) améliorent les conditions de trafic.

9-2- Un serveur de règles pour faciliter la tâche

La mise en place de la QoS reste délicate. Pour faciliter cette tâche, le serveur de règles (Policy Server) est une solution de choix. Il permet de paramétrer la QoS sur tout le réseau à partir d'une seule machine, en indiquant aux commutateurs, aux routeurs, etc., comment se comporter avec les flux qu'ils traitent. Pour cela, deux approches sont possibles : le Cops (Common Open Policy Services), qui assure le dialogue entre un serveur centralisant les règles et les équipements du réseau, ou le DEN (Directory Enabled Networking), qui définit une architecture basée sur un annuaire centralisé.

Pour rappel, dans notre projet, le but n'est pas d'implémenter ou de créer un nouveau mécanisme de QoS, mais tous simplement d'utiliser les statistiques qui ont été obtenus à partir du sniffer pour savoir :

- Quelles sont les applications les plus utilisées dans le réseau ?
- Anticiper les goulots d'étranglement.
- Eliminer les applications qui ne correspondent pas à la stratégie d'entreprise (tel que l'application du chat).

A partir de ces informations, on peut implémenter les mécanismes de qualité de service qui existe déjà dans les équipements réseaux (Commutateurs, Routeurs) pour améliorer la bande passante et donner des priorités aux applications critiques de l'entreprise (ERP, accès aux portails etc.). De plus on peut également intégrer des applications qui sont sensible aux délais, telle que la téléphonie sur IP et la vidéoconférence.

VII- Conclusion

Ce chapitre a décrit la conception que nous avons faite pour la capture et l'analyse des trames circulant dans le réseau ainsi que les statistiques faites sur ces trames. Ensuite la détection des différentes stations actives sur ce réseau. Et puis le traitement de cartographie et la détection de dysfonctionnement par SNMP et enfin la qualité de service.
Le chapitre suivant décrira la mise en œuvre de notre application.

Chapitre V :
Mise en oeuvre

I- Introduction

Dans ce chapitre nous allons mettre en œuvre l'objectif de notre conception, qui consiste à la mise en place, de la cartographie du réseau, de la capture des trames circulant sur ce réseau, et à faire une analyse et des statistiques sur ces trames capturées et enfin établir la qualité de service.

Dans la section suivante nous allons parler de l'environnement de développement de notre application. La section 3, décrira l'implémentation de la solution choisie, suivie, par le test dans la section 4 et enfin, la validation dans la cinquième section.

II- L'environnement de développement (Implémentation)

Nous avons vu, que l'architecture de la capture des trames Ethernet que nous avons présentées dans les chapitres précédents ; est une architecture d'analyse de réseau destinée à la plate forme de Windows 32, elle est basée sur le modèle BPF, et sur la librairie de fonction Libpcap. Nous avons également vu, que cette architecture était basée sur un pilote (appelé aussi le driver ou le gestionnaire), ce dernier a ajouté à la plate forme de Windows (win 95,win98 win NT et win 2000) la possibilité de capturer et d'envoyer des paquets sur le réseau. A cet effet, ce pilote qui réalise cet ensemble de taches, doit être installé dans l'ordinateur qui est lui même relié au réseau en question, pour pouvoir dialoguer avec la carte réseau (Adaptateur réseau), et écouter le support sur lequel est connecté notre ordinateur. Nous avons vu également dans la partie concernée à SNMP qu'il faut installer le protocole SNMP pour pouvoir utiliser ces fonctions.

1- Environnement matériel

La réalisation de ce logiciel nécessite un environnement réseau local, ainsi, pour effectuer une capture de trames Ethernet circulant dans ce réseau, ou chercher un nom de post sur le réseau par exemple ; le poste ou s'exécutera l'application doit être relié à ce réseau local.

2- Environnement logiciel

2-1- Le système d'exploitation

Actuellement, la majorité des utilisateurs utilisent des PC sous Windows, et sachant que Windows offre de puissants outils et interfaces que l'on peut exploiter, notre logiciel devra fonctionner sous la forme de Win 2000.

2-2- Environnement de programmation

Tout développement de logiciel passe traditionnellement par une étape délicate :
(le choix du langage). Effectivement, afin de développer notre logiciel, il faudrait choisir le langage adéquat.
Vu que le logiciel sera développé dans un environnement réseau et ses modules vont manipuler beaucoup de pointeurs et références (comme nous avons vu sur les fonctions de la librairie libpcap qui utilisent beaucoup de pointeurs). Nous avons choisi Visual C++ (VC++), c'est un langage de programmation riche d'utilitaires qui déchargent les développeurs des taches fastidieuses, en leurs permettant de se consacrer à l'essentiel.

La fonctionnalité la plus importante de Visual C++ est le fait qu'il travaille à la fois avec la librairie MFC et le langage C++ successeur de C utilisé par l'interface de programmation des applications Windows (API).
L'API Windows –large palette de fonctions implémentée dans un jeu de librairie dynamique- apparue avec Windows, a été la base de toute programmation sous windows.
La librairie MFC (Microsoft Fondation Class) est une librairie qui comprend presque toutes les fonctions implémentées dans l'API Windows et les organise en hiérarchie de classe d'où leur simplicité d'emploi.
Windows autorise l'édition de liens dynamiques avec les modules objet d'un programme.
Cela signifie que des bibliothèques spécialement conçues à cet effet peuvent être chargées et liées à l'application (exemple : libcap.lib, pcap .lib, Packet.dll, etc...). C'est ce qu'on appelle bibliothèque dynamique DLL.

2-3- C++ contre JAVA

(Pourquoi C++ et non pas JAVA ?)
Tout d'abord, parce qu'un programme compilé sera toujours plus performant qu'un programme interprété. Ainsi, notre logiciel doit sauvegarder des informations sur le disque, alors que nous savons bien que les appels JAVA, pour des raisons de sécurité, ne peuvent pas écrire sur le disque dur. Une application C++ écrite sous Windows est plus souple, car elle peut appeler à volonté les fonctions de Win32.

2-4- Les protocoles de communications

Pour que notre application fonctionne, il faut installer le protocole TCP/IP pour communiquer avec les serveurs du réseau LAN-Ethernet, ainsi que le pilote : network Packet Capture Driver (destiné à la plate forme de Windows 95/98/ME/NT/2000/XP) pour la capture des trames, et le protocole SNMP pour la gestion de la cartographie et pour l'acheminement des informations.
Et sur les autres postes du réseau que nous voulons administrer, nous devrons installer et configurer le protocole SNMP pour la gestion de cartographie et pour l'acheminement les informations entre ces postes et l'administrateur.

2-5- Les Bibliothèques dynamiques

Notre logiciel fait appel aux bibliothèques dynamiques (DLL) de langage C++ version 6.0 donc pour pouvoir exécuter l'application, il faut que ces DLLs soient installées au préalable au niveau de la station (PC administrateur) cela se réalise par l'installation de winpcap.

3- Programmation

Cette activité consiste à passer du résultat de la conception détaillée à un ensemble de programmes ou de composants de programmes.

3-1- Quelques exemples d'utilisation des fonctions de wpcap.dll

Les exemples que nous allons présenter (opérations effectuées sur le pilote de la capture, et un dialogue avec l'adaptateur réseau) montrent comment écrire un programme de capture simplifié, dans un réseau local.
Ces exemples représentent des programmes de capture de paquets très simples, qui montrent l'utilisation du Packet Capture Driver à travers L'API wpcap.dll.

Nous allons présenter quelques échantillons de programmes écrits en langage C++ qui se présente comme suit :

Programme 1 : Initialisation (programme de sélection de l'adaptateur réseau)

Il va nous falloir choisir une interface à sniffer. Pour cela, nous avons deux solutions : soit nous la choisissons "manuellement" et nous mémorisons, soit nous faisons un "autodetect" avec la fonction **pcap_lookupdev()**.
La valeur de retour est un pointeur sur la première interface réseau détectée ou NULL si une erreur est survenue.

```
//***programme de sélection de l'adaptateur réseau***
#include <pcap.h>
#include<iostream.h>
#include <stdio.h>
.....
char *dev ; // C'est un pointeur sur l'interface réseau détectée
   dev = pcap_lookupdev(errbuf);
   printf("Device: %s\n", dev);
.....
```

Programme 2: Permet l'ouverture de l'adaptateur réseau et l'initialisation en mode Promiscous après l'avoir sélectionné.
Après avoir choisit l'interface de réseau, il va nous falloir un descripteur pour la capture des paquets. On l'obtiendra par la fonction **pcap_open_live(),** dont la syntaxe est la suivante :

pcap_open_live(char *device, int snaplen,int promisc, int to_ms, char *ebuf);

```
// Ouverture de l'adaptateur réseau sélectionné au préalable, et l'initialisation en mode
Promiscous
.....
pcap_t *desc = NULL; // C'est le descripteur de la capture
if ((desc= pcap_open_live(dev, 65536, 1, 0, errbuf))== NULL)
 {cout<<"ggggggggggggg";}
 else
        { exite(1);}

// Le 0 indique le mode Promiscous
// 65536 indique la taille de packet à capturer
.....
```

Apres avoir obtenu notre descripteur, il va nous falloir le numéro et le masque de réseau associé à l'interface que l'on désire sniffer. On peut l'obtenir avec la fonction **pcap_lookupnet()**. Dont la syntaxe est :

pcap_lookupnet(char *device, bpf_u_int32 *netp,bpf_u_int32 *maskp, char *errbuf);

Programme 3 : Définir le protocole de base :
Cela se réalise en utilisant la fonction **pcap_datalink ()** ; pour savoir où se trouve le protocole de la couche Internet (son emplacement par rapport au protocole de la premier couche).
.....

```
int ethhdr; // C'est le nombre d'octets à décaler pour trouver le protocole de la
//2emme couche
switch (pcap_datalink (desc))
{  case DLT_NULL:
     printf ("Data Link Type: NULL\n");
     ethhdr = 4;
     break;
   case DLT_EN10MB:
     printf ("Data Link Type: Enthenet 10MB\n");
     ethhdr = 14;
     break;
   case DLT_EN3MB:
     printf ("Data Link Type: Enthenet 3MB\n");
     ethhdr = 14;
     break;
   case DLT_PPP:
     printf ("Data Link Type: PPP\n");
     ethhdr = 4;
     break;
   case DLT_SLIP:
     printf ("Data Link Type: SLIP\n");
     ethhdr = 16;
     break;
   case DLT_FDDI:
     printf ("Data Link Type: FDDI\n");
     ethhdr = 21;
     break;
   case DLT_RAW:
     printf ("Data Link Type: RAW\n");
     ethhdr = 0;
     break;
   default:
     printf ("Dispositivo não suportado\n");
     break;    }
.....
```

Programme 4 : Capturer des trames.
La capture des trames se fait grâce à la fonction **pcap_next()** ;
Le résultat (trame capturée du réseau) est mis dans un buffer ; pour des traitements ultérieurs.
Le principe est le suivant:
1. lancer la fonction **pcap_next()** ;
2. mettre le résultat dans le buffer ;
3. ***si*** on veut arrêter le processus de capture **alors** allez à 4 **sinon** allez à 1.
4. fin ;

```
// Programme de capture des trames
u_char *packet; // C'est la trame capturée
while (-1)
// Boucle sans fin
// On attend de recevoir plein de paquets, que l'on doit recevoir
```

```
// Pendant cette boucle. Il lira le réseau sans arret.
{
packet = (u_char *) pcap_next(desc, &usefull);
// On lit le prochain paquet. S'il n'y a pas de paquets en attente, alors
// La fonction renvoi NULL (et au bout d'une seconde, 1000 ms (notre timeout)
// Sinon, elle renvoi l'adresse du paquet reçu et on le traite après:
if (packet != NULL)
{   buffer = packet ;
}
}
```

Programme 5 : Décoder les trames
Les trames capturées ont la forme de ASCII, ce programme fait la conversion du code ASCII vers le décimal.

Programme 6 : Analyser les trames capturées
Cela ce fait en deux étapes:

- La première (dans le protocole de la couche Internet): il faut détecter le protocole de la couche transport.
- La deuxième (dans le protocole de la couche transport) : il faut détecter le numéro de port de la couche application utilisé.

Programme 7 : Faire les statistiques:
Faire des statistiques sur les protocoles de la couche application déjà capturée par le programme 4.

Programme 8 : Programme de fermeture de l'adaptateur :

```
.....
// Fermer l'adaptateur réseau et sortir
pcap_close(desc);
.....
```

III- Test du logiciel

L'objectif du test d'un logiciel est de détecter les erreurs. La mise au point a pour but de localiser ces erreurs et de les corriger.
Le test permet de réaliser des contrôles pour la qualité du système. Il s'agit de relever les éventuels défauts de conception et de programmation.
Pour tester notre logiciel, notre chois d'application est fait sur la gestion de réseau local Ethernet.
Dans cette section nous allons présenter des copies d'écran de notre logiciel, cette section peut être considérée comme un manuel d'utilisation de notre logiciel puisque nous allons passer par presque toutes les étapes d'utilisation.

Lors de l'ouverture du logiciel, une interface contenant un menu va apparaître.
L'utilisateur peut ouvrir ou fermer un logiciel conçu par le concepteur.

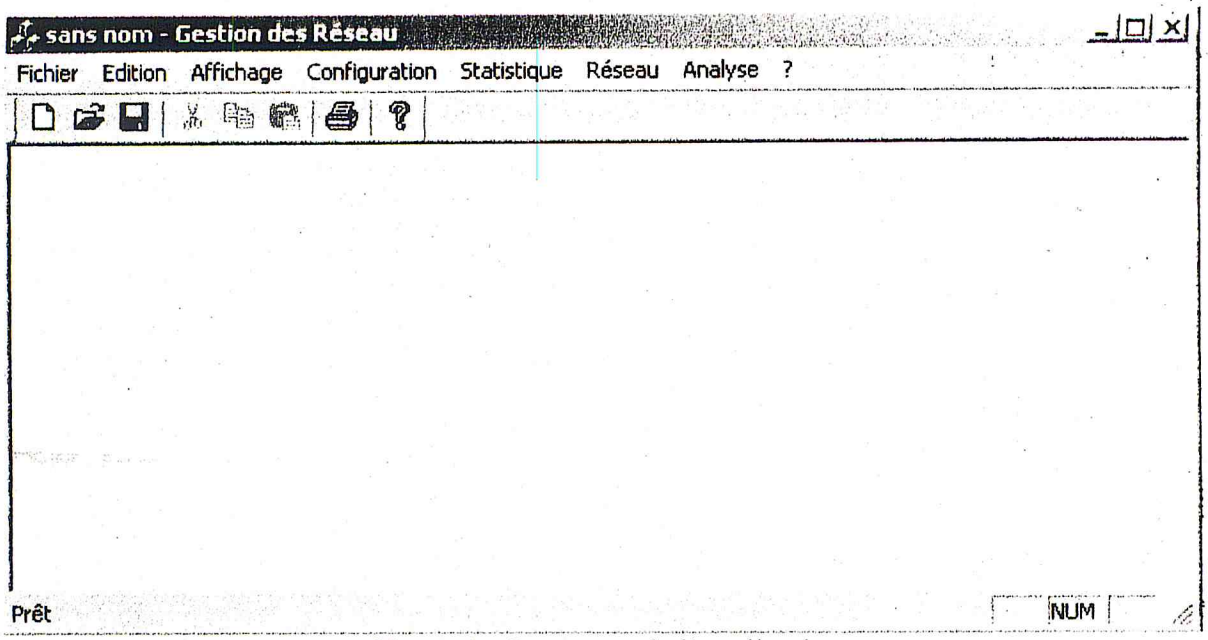


Figure 1 : Ouverture de l'application

Cette figure montre notre application lors de son ouverture.
Le menu Fichier permet l'ouverture, la fermeture, l'enregistrement de l'application.
Le menu Edition permet de copier, couper, coller des textes.
Le menu Affichage permet d'afficher la barre d'outils ou bien la barre d'état.

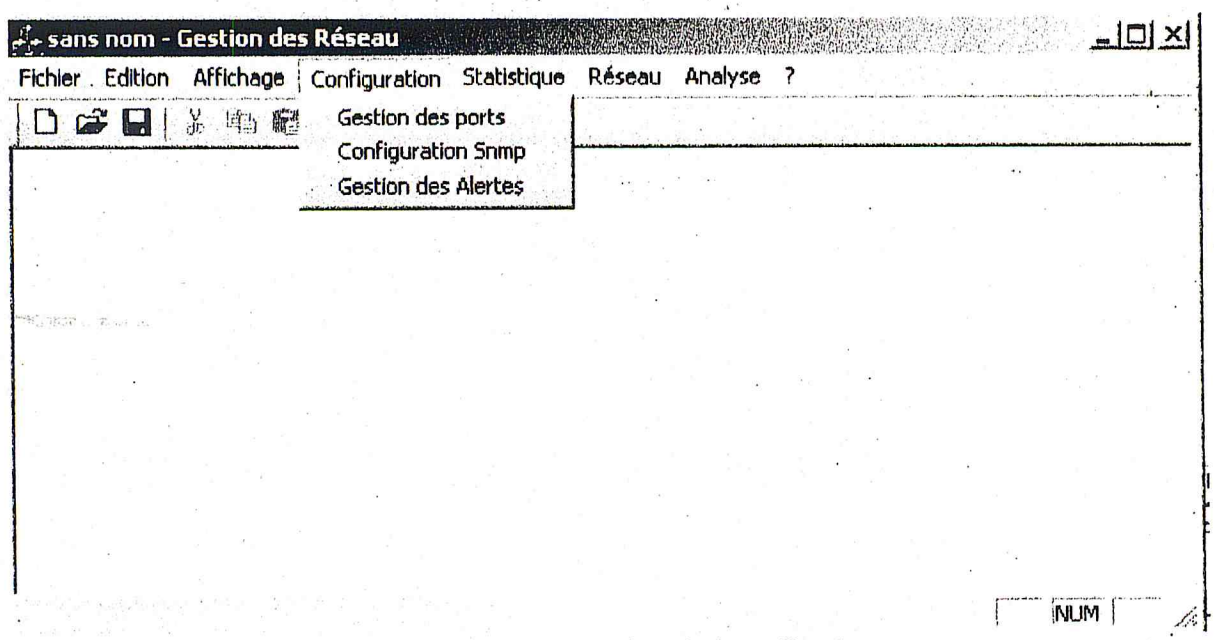


Figure 2 : Configuration de l'application

Le menu Configuration permet de faire :
Une gestion des ports, une configuration d'SNMP et la gestion des alertes.

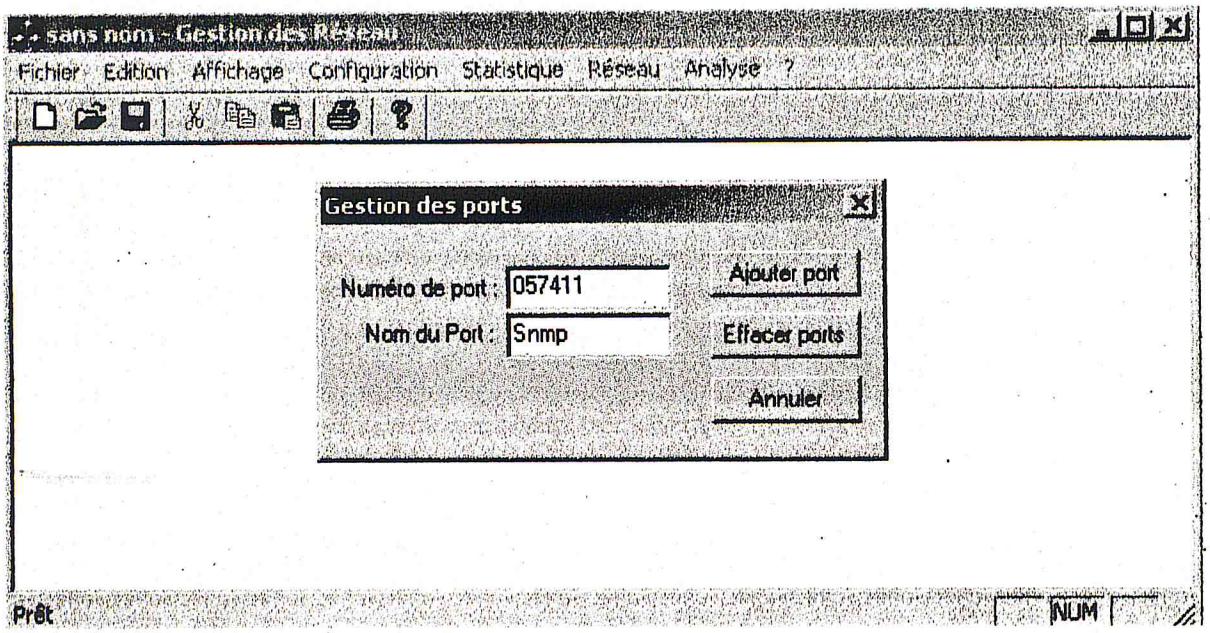


Figure 3 : Gestion des ports

Cette figure nous montre la possibilité d'attribuer un nom à un numéro de port.

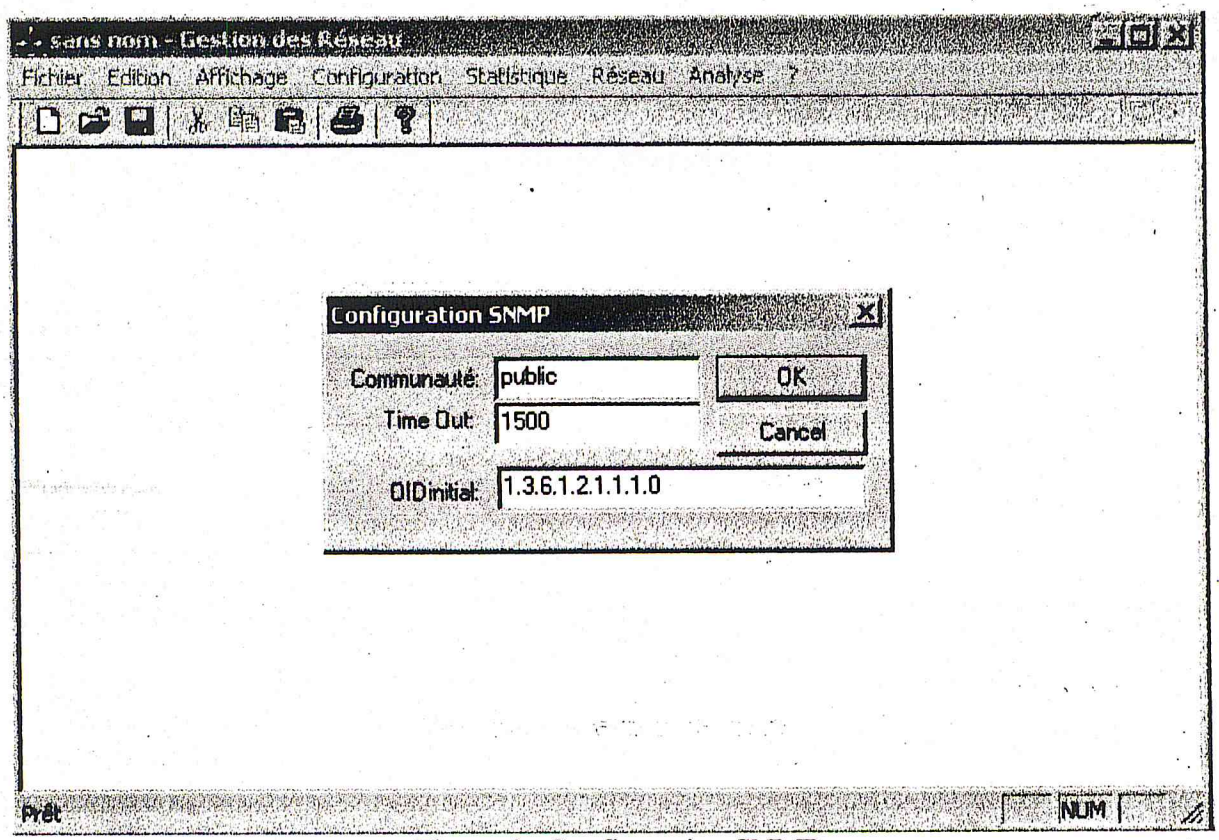


Figure 4 : Configuration SNMP

Le nom de la communauté de Configuration SNMP est le même que celui de la cartographie. Nous pouvons modifier les informations associées au poste lors de son affichage par OIDinitial.

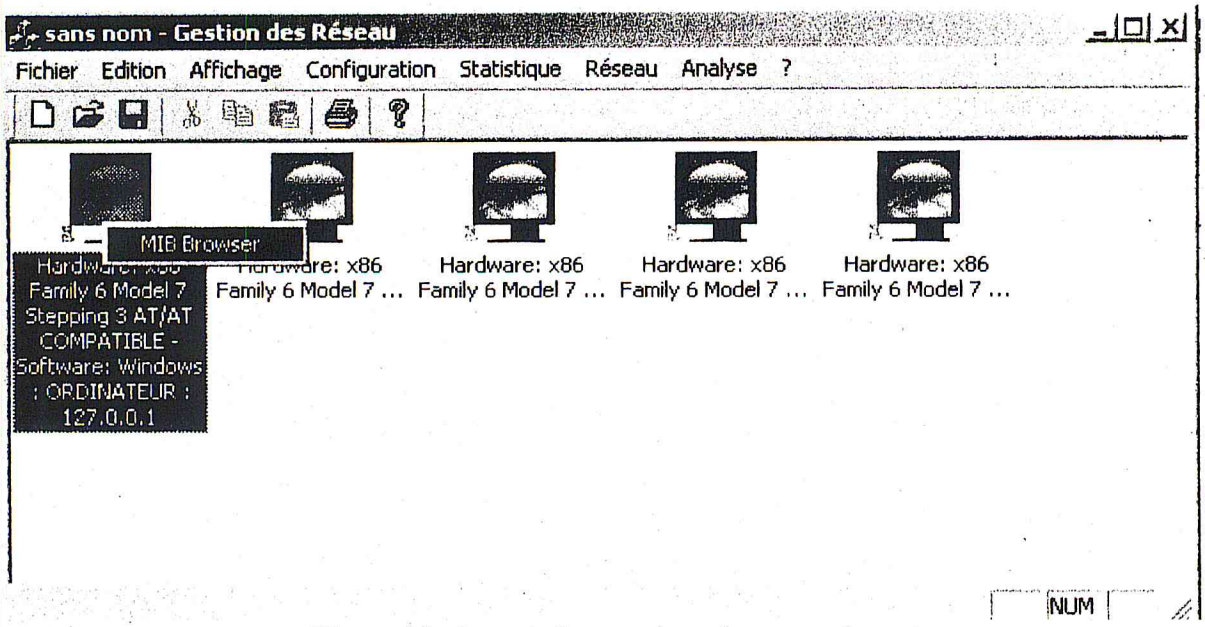


Figure 11 : Les stations actives de notre réseau

La figure montre la MIB Browser, en cliquant par le bouton droit sur une des stations actives.

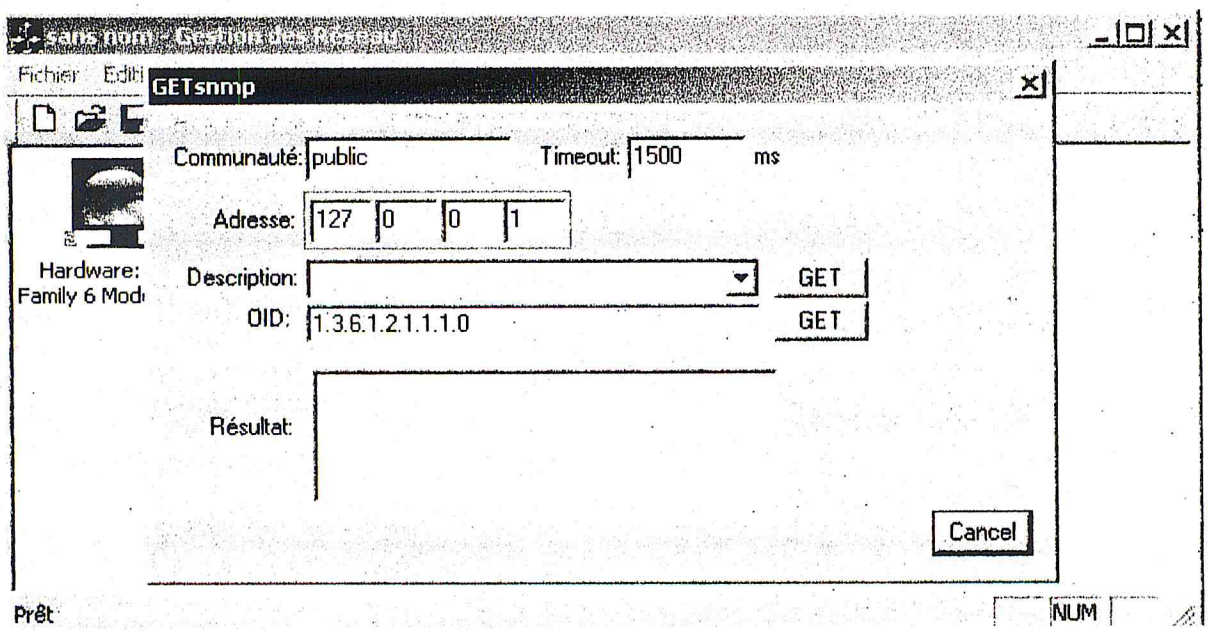


Figure 12 : SNMP

Cette figure montre la commande Get Snmp, pour une demande de lecture d'un objet se trouvant dans l'arbre hiérarchique de la MIB.
L'adresse par défaut est celle du premier poste active trouvé.
Nous pouvons accéder à cette commande soit en cliquant sur, MIB Browser, soit dans le menu réseau, sur SNMP.

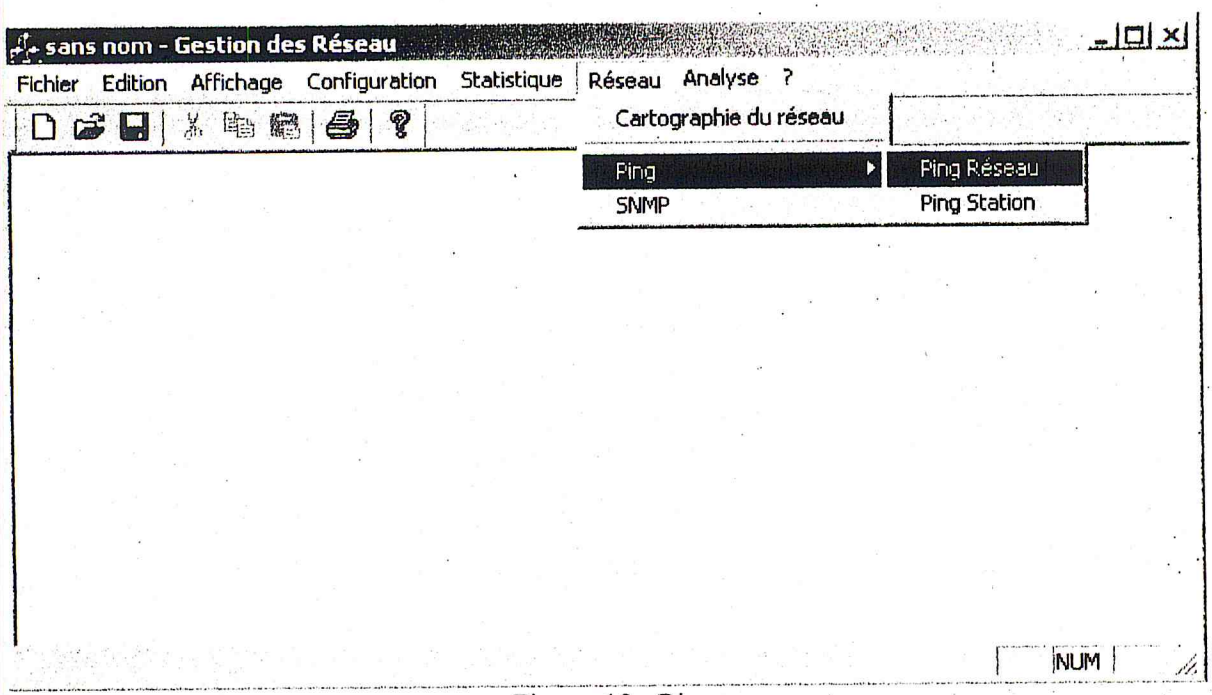


Figure 13 : Ping

Nous pouvons faire un ping par réseau ou par station.

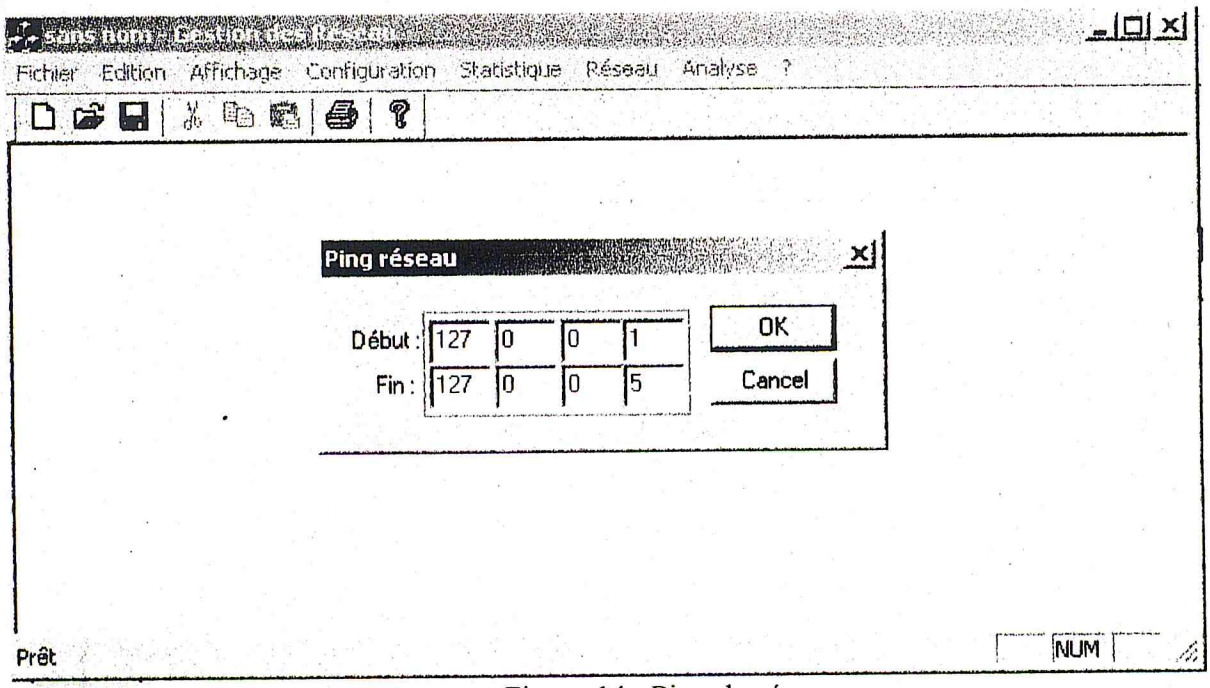


Figure 14 : Ping du réseau

Le Ping réseau se fait sur une plage d'adresse, qui est la même que celle de la cartographie.

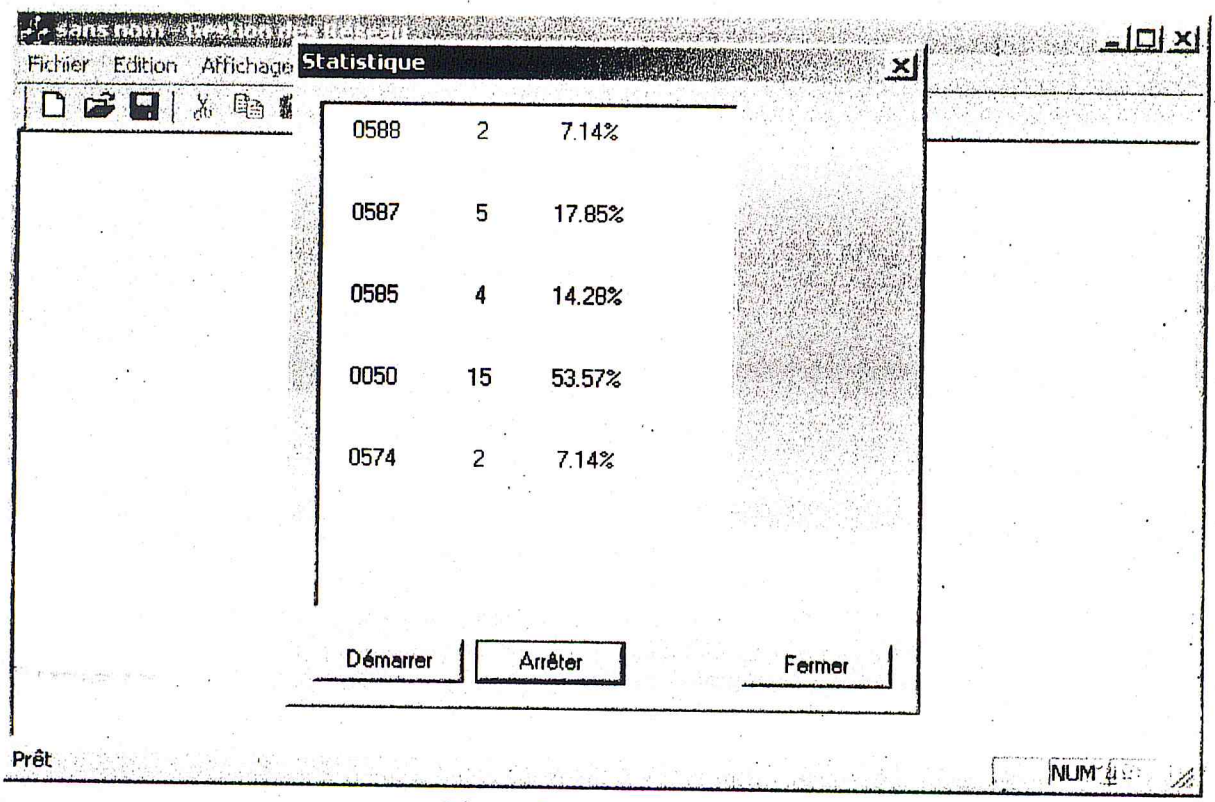


Figure 7 : Les statistiques

Cette figure montre les statistiques obtenues après avoir lancer la procédure.
L'utilisateur peut la démarrer, l'arrêter ou bien la fermer.

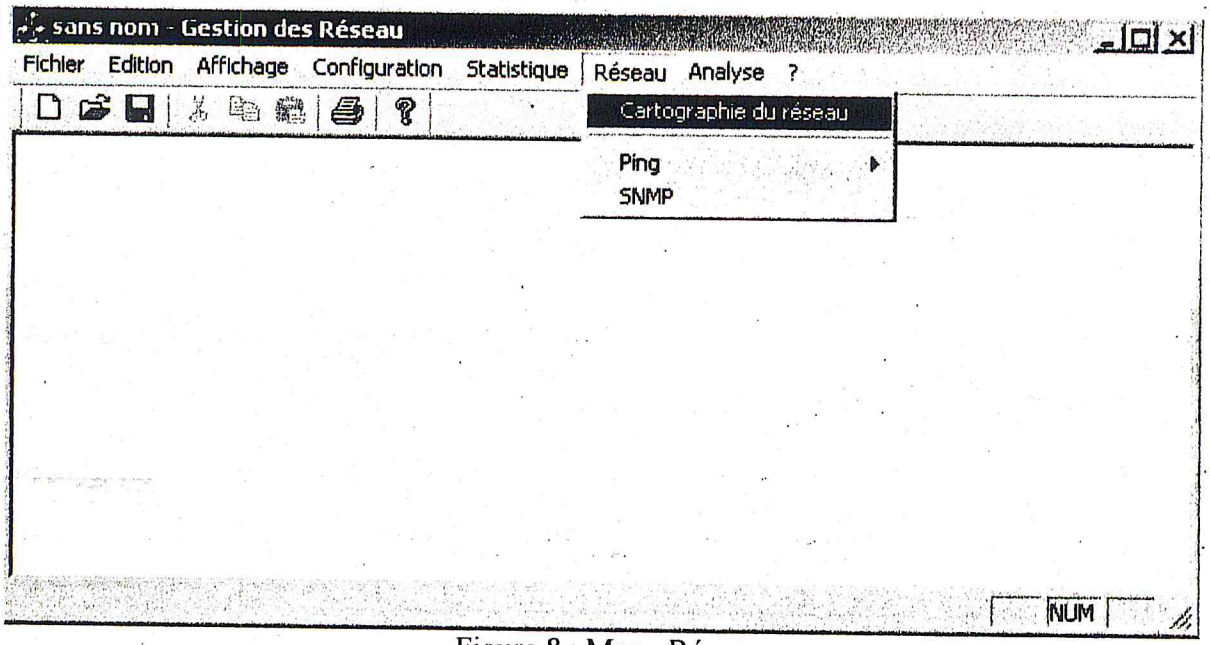


Figure 8 : Menu Réseau

Cette figure montre le menu Réseau, qui permet de donner la cartographie du réseau, le ping et Snmp.

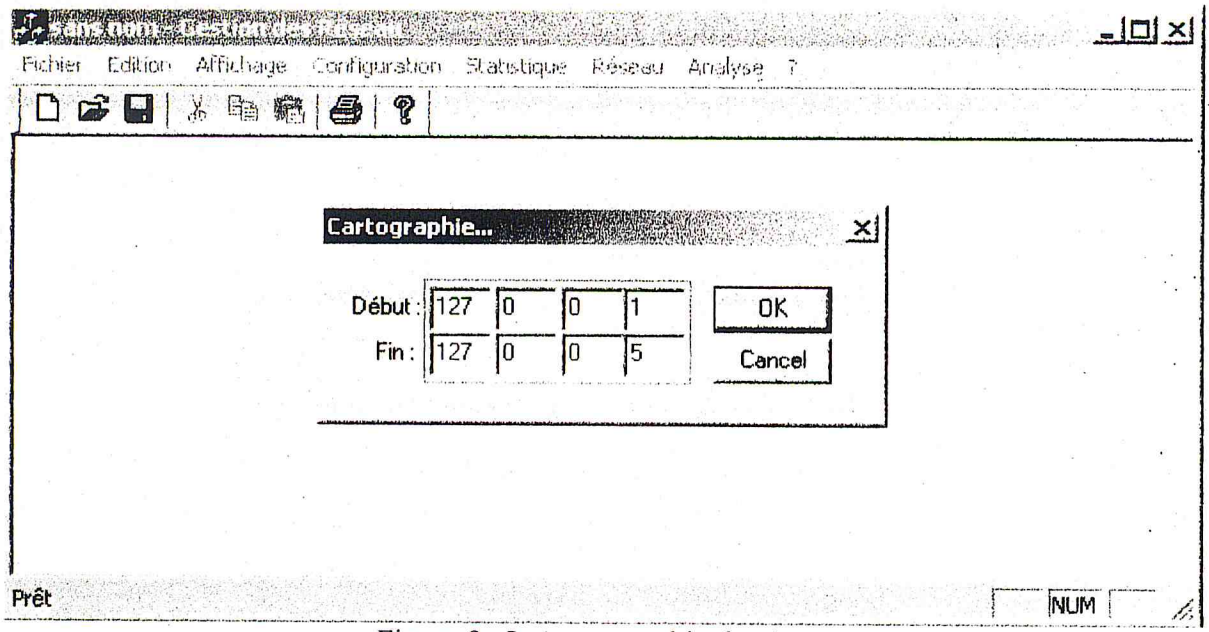


Figure 9 : La cartographie du réseau

Cette figure montre la cartographie du réseau sur une plage d'adresse obtenu par la capture, en spécifiant une adresse début et une adresse fin.
On valide avec OK pour afficher toutes les stations actives ou bien on quitte avec Concel.

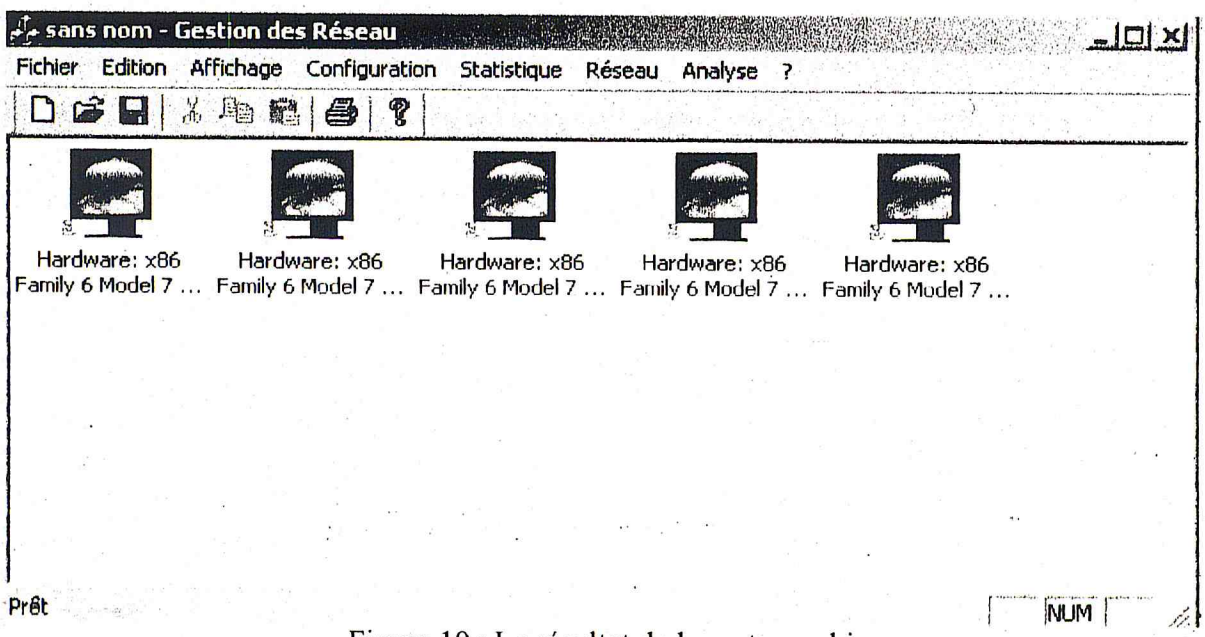


Figure 10 : Le résultat de la cartographie

Cette figure montre les différentes stations actives au moment du lancement de la cartographie.

VI- Les modes de connexion

1- Mode avec connexion (Orienté connexion)

Dans ce mode toute communication entre deux équipements fait appel à trois phases distinctes :
- Etablissement de la connexion (par l'envoi d'un bloc de données spécial, elle peut être refusée et la communication n'aura pas lieu, ou acceptée, elle sera établie par mise en place d'un circuit virtuel dans le réseau reliant l'émetteur au récepteur).
- Transfert de données, qui est la phase durant laquelle effectivement les données de l'utilisateur sont transportées d'une entité à l'autre.
- Libération de la connexion.

C'est le fonctionnement bien connu du réseau téléphonique classique.

Avantages :
- La sécurisation du transport par identification claire de l'émetteur et du récepteur.
- La possibilité d'établir à l'avance des paramètres de qualité de service qui seront respectés lors de l'échange des données.

Inconvénients :
- La lourdeur de la mise en place de la connexion qui peut se révéler beaucoup trop onéreuse si l'on ne veut échanger que quelques octets.
- La difficulté à établir des communications multipoint, dans ce cas il faut établire autant de connexions que des points à atteindre.

2- Mode sans connexion (Non orienté connexion)

Dans le mode sans connexion les blocs de données, appelés datagrammes (contenant le nom et l'adresse du destinataire), sont transférés sans établire une connexion et sans vérifier à l'avance si l'équipement à atteindre, ainsi que les nœuds intermédiaires sont bien actifs. C'est alors aux équipements gérant le réseau d'acheminer le message étape par étape en assurant éventuellement sa temporisation jusqu'à ce que le destinataire soit actif (le message est mis dans une boîte de réception associée à l'utilisateur à atteindre). Ce service est celui du courrier postal et du courrier électronique.

Avantage : la rapidité de transmission, mais il reste pour les courts messages.

Inconvenant : il n'assure aucune fiabilité de transmission.

VII- L'adressage (Logique et Physique)

1- Définition de l'adresse physique

Elle est inscrite **en dur** sur la carte réseau (carte Ethernet), qui a une adresse unique, adresse physique, faite de deux parties:
- Les 3 premiers octets d'une adresse Ethernet, dite aussi **adresse MAC**, identifient le numéro du constructeur de la carte.
- Les autres 24 bits identifient le numéro de la carte.

Le numéro de téléphone est un exemple classique d'adressage physique. Les adresses Transpac sont aussi des adresses physiques. Deux adresses particulières sont réservées :
FF: FF: FF: FF: FF:FF (c'est l'adresse broadcast ou adresse de diffusion) & 0:0:0:0:0:0

2- Définition de l'adresse logique

C'est **l'adresse IP** (adresse Internet), fournit de façon définitive par le SysOP du réseau ou à la demande par le fournisseur d'accès Internet, l'ISP. Les adresses IP ont une longueur fixe de 32 bits, soit 4 octets. Elles se composent de deux parties :

- L'adresse du réseau.
- L'adresse de l'hôte sur ce réseau.

Une adresse IP doit être unique au monde, c'est le NIC (Network Information Center) qui attribue les adresses de réseau. L'adresse de l'hôte est laissée à l'appréciation de l'administrateur du site. L'écriture des adresses IP se fait octet par octet, séparée par un point. L'adresse IP est responsable de l'établissement d'une connexion logique entre une source et une destination sur un réseau.

Un exemple classique d'adresse logique est l'adresse e-mail, contenant un nom local d'abonné et un nom global de domaine.

VIII - Système en couches

1- L'intérêt d'un système en couche

A fin de réduire la complexité de conception, et séparer le problème en différant parties, la plupart des réseaux sont organisés en séries de couches ou niveaux, chacune étant construite sur la précédente. Le nombre de couches, leur fonction, varient selon les réseaux.

Dans chaque réseau, le but de chaque couche est d'offrir certains services aux couches plus hautes, en leur épargnant les détails de la mise en œuvre de ces services.
Entre autre des interfaces bien conçues facilitent le changement de mise en œuvre dans une couche (par exemple le remplacement des lignes téléphonique par des canaux de satellites), il suffit à la nouvelle installation d'offrir exactement à sa voisine du dessus le même ensemble de services que l'ancienne.
De plus la Facilité d'implémenter ou de changer un ou plusieurs protocoles, par exemple : si on veut changer le protocole IP, on change que le protocole de la couche IP.

L'intérêt d'un système en couche est de faciliter le développement et de simplifier l'apprentissage et de standardiser les interfaces.

2- Concept d'un système en couche

Le concept important est qu'il faut considérer que chaque couche est programmée comme si elle est vraiment horizontale, c'est à dire qu'elle dialogue directement avec sa couche paire réceptrice grâce au protocole de la couche. Au moment de dialoguer avec sa couche paire, chaque couche rajoute un en-tête et l'envoie (virtuellement, grâce à la couche sous-jacente) à sa couche paire.

Conclusion

Dans cette annexe nous avons fait un survol sur les concepts de base et les terminologies du domaine de réseau. Nous avons défini les sous réseaux ainsi que les systèmes autonomes. Nous avons également parlé des types et topologies des réseaux, des raccordements physiques et des équipements téléinformatiques existants.
Ensuite nous avons présenté les différents modes de connexion et défini l'adressage logique et physique et enfin nous avons expliqué l'intérêt et le concept d'un système en couche.

Annexe B

Dans cette annexe, nous présenterons les Sockets, que nous avons utilisé pour la réalisation de notre application.

I- Introduction

Les Sockets (prises de raccordement) offrent aux programmeurs une interface entre le programme d'application et les protocoles de communication.
Une socket est définit par une adresse réseau (IP) et un numéro de port, utilisé pour envoyer ou recevoir des données.

II- Les services Socket

Il existe plusieurs services pour l'établissement et la transmission des données à travers l'interface Socket en fonction du mode de transmission choisi.
Les services de base pour la communication entre deux processus distant sont les suivantes :

- Socket() : Création d'une Socket.
- Bind() : Liaison d'une Socket à une adresse locale
- Sendto () : Envoyer des données sur la Socket.
- Recvfrom () : Recevoir des données sur la Socket.
- Listen () : Gérer les requêtes de connexion.
- Connect () : Se connecter à une Socket distante.
- Colose() : Fermer la connexion.

III- Les types de socket

Le type de Socket est choisi lors de sa création, on distingue essentiellement trois types :

- **Le type SOCK_DGRAM** : elles sont destinées aux communications en mode non connecté. Le protocole approprié dans le domaine Internet est le protocole UDP.
- **Le type SOCK_STREAM** : elles sont destinées aux communications en mode connecté, le protocole approprié dans le domaine Internet est le protocole TCP.
- **Le type SOCK_Raw** : il permet l'accès aux protocoles de plus bas niveau tel que le protocole IP. Son usage est réservé aux super utilisateurs, il permet donc d'implanter de nouveaux protocoles de transports.

1- La communication en mode Datagramme

Ce type de communication permet à une application d'envoyer des messages à une autre en mode non connecté. Dans ce mode il n'y a ni établissement d'une connexion entre les deux systèmes, ni contrôle de flux, ni reprise sur erreur (retransmission). Le protocole utilisé est UDP.
Les principales caractéristiques de la communication entre processus à l'aide des Socket sont :

- Lorsqu'un message est envoyé sur une Socket à destination d'une autre, l'expéditeur n'obtient aucune information sur l'arrivée de son message.
- Les limites des messages sont préservées.

Le manque de sûreté dû à la première caractéristique est relatif, puisque les applications utilisant UDP peuvent implanter leurs propres politiques de retransmission, de Time-out et du contrôle de flux.

Principe général :

L'échange de datagrammes est réalisé par l'intermédiaire des Socket du type **SOCK_DRAM.**

Un processus souhaitant émettre un message à destination d'un autre doit d'une part disposer d'un point de communication local (descripteur de Socket local) et d'autre part connaître l'adresse de la Socket appartenant à son interlocuteur.

L'utilisation des Socket en mode datagramme :
Les étapes nécessaires pour un dialogue entre deux processus, un client et un serveur :

Le code du serveur :
- Création et attachement d'une Socket.
- Boucle infinie dans laquelle le serveur :
  - Attend une requête.
  - Traite la requête.
  - Crée une réponse.
  - Envois la réponse.

Le code du client :
A. Création de la Socket locale.
B. Préparation de l'adresse du serveur.
C. Envoi du message.
D. Attente de la réponse.
E. Si timer expiré aller à 'C'.
F. Si réponse arrivée alors exploiter le résultat.

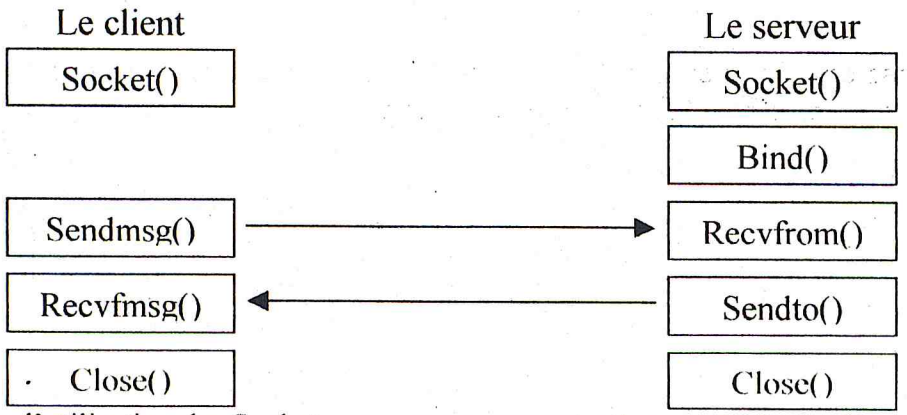


Figure 1 : l'utilisation des Sockets pour une communication en mode datagramme

2- La communication en mode connecté

Dans le mode connecté, une connexion est établie avant le transfert des données, cette connexion reste la même pendant toute la durée de vie de la communication, avec des contrôles nécessaires pour la fiabilité de la liaison. Le protocole concerné est TCP.
Les principales caractéristiques de ce mode de communication sont :
- La fiabilité de la transmission
- La préservation de l'ordre des données
- La non-duplication des données

Principe général :
L'échange des données est réalisé par l'intermédiaire des Sockets du type **SOCK_STREAM.** Pour utiliser ce mode de communication, il faut établir une connexion point à point. L'établissement de la connexion dans le mode connecté met en lumière la dissymétrie entre les deux processus impliqués. L'un des deux processus (le client) demande à l'autre(le serveur) s'il accepte la connexion.

Listen gère les problèmes inhérent à cette approche en établissant une queue de réception de requêtes.
La fonction **Listen** laisse la Socket accepter les requêtes de connexion qui sont placées dans la queue pour traitement ultérieur.
Le format de fonction est :

Listen(Socket, Longeur_queue)

Le serveur utilise alors la fonction **Listen** pour gérer les requêtes de connexion et la fonction **Accept** pour attendre une connexion.
Le format de la fonction Accept est :

Accept(Socket, Adresse, Longeur)

7- Récupérer des informations de statut

Plusieurs fonctions sont utilisées pour obtenir des informations relatives à une connexion.
Elles peuvent être utilisées à tout moment, bien qu'on les emploie le plus souvent pour vérifier l'intégrité d'une connexion en cas de problème.

Les fonctions de statut ont besoin de connaître le nom de la connexion locale, et renvoient un ensemble d'informations qui peut comprendre les noms des sockets locales et distantes, le nom de connexion locale, l'état des fenêtres d'émission et de réception, le nombre de tampons en attente d'acquittement, le nombre de tampons attendant des données, les valeurs courantes des variables urgentes, précédence, security et timeout.

La fonction **Getsockopt** permet à une application d'interroger une socket.
Deux autres fonctions de statut, donnent des infirmations relatives à l'adresse locale de la Socket. La fonction **getpeername** renvoie l'adresse de l'extrémité distante.
La fonction **getsockname** renvoie l'adresse locale d'une socket.
Le format de ces fonctions est le suivent :

Getpeername(socket, Adresse_destination, Longeur_adresse)

Getsockname(socket, Adresse_locale, Longeur_adresse)

8- Fermer une connexion

La fonction **Close** ferme une connexion, elle a besoin uniquement du nom de connexion locale pour s'exécuter. Son format est le suivant :

Close(Socket)

Le nom de socket est obligatoire si une application se termine de façon anormale, le système ferme toutes les Sockets ouvertes avant la fin du processus.

Adresse_locale est l'adresse locale à laquelle se lier,
Longeur_adresse est un entier qui donne la longueur de l'adresse en octets.

3- Connexion à la destination

Après l'appel à la fonction Bind, on est dans un état non connecté. Pour passer à l'état connecté on doit ajouter la Socket de destination.
Les protocoles non connectés comme, UDP sont prêt à permettre le trafic entre les sockets sans avoir à respécifier l'adresse de destination à chaque fois. Par contre, les protocoles en connexion comme TCP imposent que les deux extrémités de la connexion soient spécifiées.
Pour établir une connexion vers une Socket distante on utilise la fonction Connect. Le format de la commende Connect est :

Connect(Socket ,Adresse_destination, Longeur_adress)

Où :
Socket est la valeur entière de la Socket à laquelle se connecter
Adresse_destination décrivent l'adresse de destination
Longeur_adresse est la longueur de l'adresse de destination, on octet.

La façon dont fonctionne la fonction Connect dépend du protocole utilisé. Pour TCP, Connect établit la connexion entre les deux extrémités et renvoi les informations relatives à la Socket de destination. S'il est impossible d'établir une connexion, un message d'erreur est généré.
Pour un protocole non connecté comme, UDP la fonction est nécessaire mais elle se contente de socket, 'adresse de destination.

4- Emettre des données

L'API comporte cinq fonctions pour l'envoi des données à travers une Socket .Ce sont les fonctions **Send, Sendto, Sendmsg, Write et Writev.**
Les fonctions **Send Write** et **Writev** ne fonctionnent qu'en mode connecté puisqu'on ne peut pas leur indiquer l'adresse de destination.
Les fonctions **Sendto** et **Sendmsg** permettent à une application d'envoyer des messages à travers une Socket non connecté, elles ont, toutes deux besoin de l'adresse de destination.

5- Recevoir des données

Puisqu'il y a cinq façons d'envoyer des données il y a cinq façons d'en recevoir : **Read, Readv, Recv, Recvfrom et Recvmsg.**
La fonction **Read** est simple et la plus utilisée dans le cas des Sockets connectées.
Son format est :

Read(Socket, Buffer, Longueur)

La fonction **Recv** peut également être utilisée avec des Sockets connectées.
Les fonctions **Recv** et **Recvfrom** permettent de lire des données depuis une Socket non connecteé. Leur format comprend l'adresse de l'émetteur.

6- Ecouter des serveurs

Une application serveur qui attend que des clients lui envoient des requêtes, doit créer une socket, la lier à un port (à l'aide de **Bind**), puis attendre les requêtes entrantes. La fonction

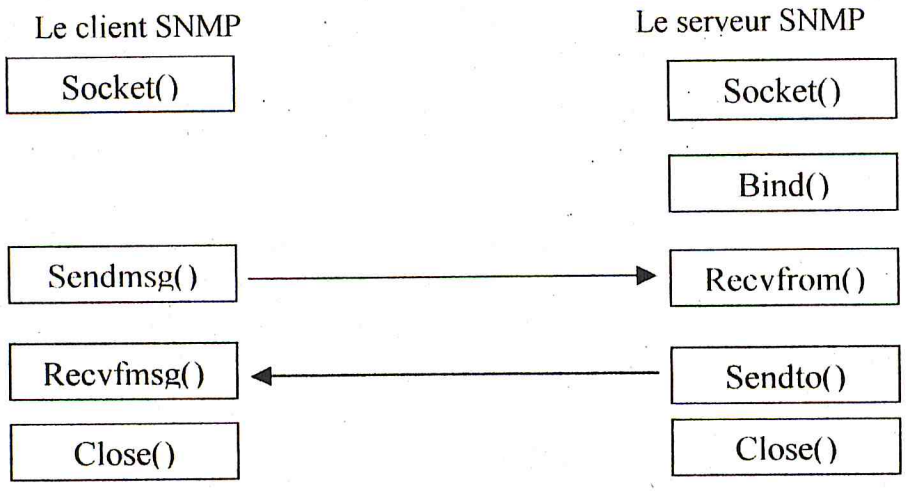


Figure3 : Le mode de communication du protocole SNMP

IV- Les fonctions des sockets

1- Créer une Socket

Cette API permet à un utilisateur de créer une Socket par un simple appel à la fonction. la fonction a besoin de :
La famille de protocoles à utiliser(afin de savoir le type de Socket assigner et comment décoder les informations associées à la Socket). Parmi les familles possibles, on trouve TCP/IP(AF_INET), AppeTalk (AF_APPLE_TALK), et le système de fichier UNIX(AF_UNIX).
Du type de communication à utiliser, ce peut être un service de Datagrammes sans connexion (SOCK_DREAM), un service à flux(SOCK_STREAM) ou un type brut(SOCK_RAW)
Du protocole spécifique.
L'appel de cette fonction se fait comme suit :

Socket (Famille, Type, Protocole)

Le retour de cette fonction est un entier qui décrit de façon unique une Socket (descripteur de la Socket).

2- Lier une Socket

Sous TCP/IP, la fonction **Socket** ne donne pas le numéro de port local, pas plus que le port de destination ou l'adresse IP de destination. La fonction **Bind** est utilisée pour spécifier l'adresse du port local de connexion (pour établir complètement la connexion).
Certaines applications (spécialement sur un serveur) utilisent un port spécifique pour une connexion. Pour, la couche transport alloue elle-même un port. On peut demander un port spécifique en utilisant la fonction **Bind**. Si le port ne peut pas être alloué (il peut être utilisé au moment même), le code de retour indique une erreur d'affectation de port.
La fonction **Bind** possède la syntaxe suivante :

Bind (Socket, Adresse_locale, Longueur_adresse)

Où :
Socket est la valeur entière, c'est un identificateur de Socket à lier,

L'utilisation des Sockets en mode Stream(mode connecté)

Les étapes nécessaires pour un dialogue entre deux processus, un client et un serveur.

Le code du serveur :

A. Création et attachement d'une Socket
B. Boucle infinie dans laquelle le serveur :
   - Attend une demande de connexion.
   - Accepte la connexion.
   - Attend les messages.
   - Envoi la réponse.
   - Aller à 'B'.

Le code du client :

A. Création de la Socket locale.
B. Connexion au site distant.
C. Envoi du message.
D. Attente de la réponse.
E. Si timer expiré aller à 'C'.
F. Si réponse arrivée alors exploiter le résultat.

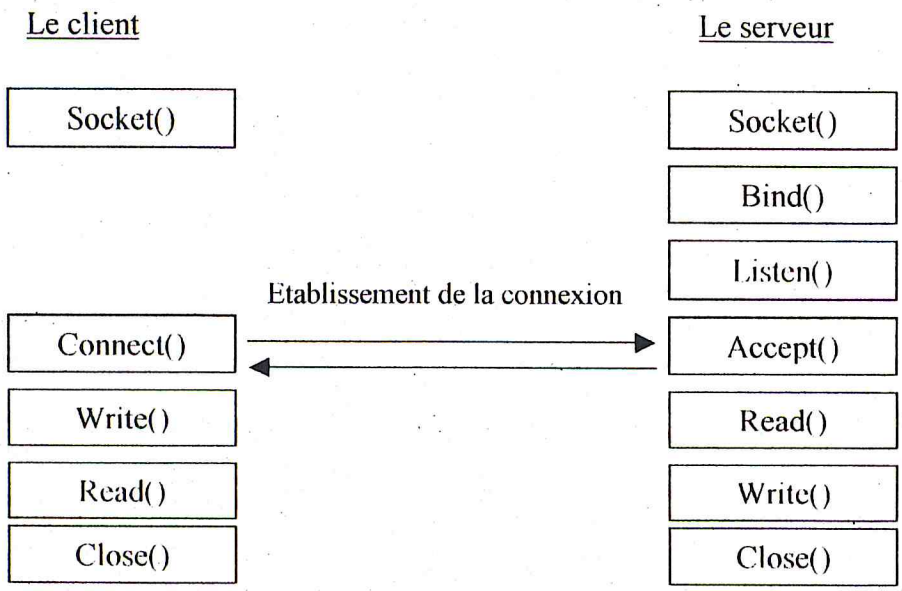


Figure 2 : Exemple d'utilisation des Sockets pour une communication en mode connecté

3- Le mode de communication du protocole SNMP

Le service de transport requis par SNMP est en mode datagramme, UDP. Cela est dû au fait que les foncions de gestion de réseau interviennent souvent dans un contexte de recherche de panne. L'application administrative est donc la mieux placée pour déterminer les contraintes de fiabilité à appliquer au trafic de gestion. Ce choix permet aux stations de gestions de déterminer le niveau de retransmission nécessaire pour s'adapter à des réseaux peu fiables ou congestifs.

Annexe C

Nous présenterons, dans cette annexe les objets de la MIB de SNMPv1 ainsi que ceux des MIBs USM et VACM définis pour SNMPv3.

I- Les définitions d'objets de la MIB de SNMPv1

La MIB contient une centaine d'objets rangés par groupes fonctionnels au nombre de dix :

- **System**
- **Interfaces**
- **Adress translation**
- Internet Protocol(**IP**)
- Internet Control Message Protocol(**ICMP**)
- Transmission Control Protocol(**TCP**)
- User Datagramm Protocol(**UDP**)
- **Egp**
- Transmission
- Simle Network Management Protocol(**SNMP**)

Ces groupes permettent de gérer uniquement un réseau TCP/IP

1- Le groupe system (1.3.6.1.2.1.1)

Il contient des informations générales sur le système administré.
Sysdescr : Une description textuelle de l'entité. Cette valeur doit inclure le nom complet et l'identification de version du type hardware, du logiciel du system d'exploitation et du logiciel de networking du système. De type DisplayString (size (0..255)
SysObjectID: Identificateur du fabriquant. Cette valeur est allouée à l'intérieur du sous-arbre SMI (entreprise) et fournit une signification simple et non ambiguë pour déterminer quelle sorte de société est administrée. De type Object IDENTIFIER.
SysUpTime: Durée en centième de secondes depuis l'initialisation de l'agent. De type TimeTicks.
SysContact: L'identification textuelle de la personne qui sert de contact pour ce nœud administré ainsi que les renseignements pour la contacter. De type DisplayString (size (0..255).
SysName: : Un nom assigné administrativement pour ce nœud administré. Par convention c'est le nom de domaine. De type DisplayString (size (0..255).
SysLocation: Description de l'emplacement physique de l'équipement. De type DisplayString (size (0..255).
SysServices: Numéro de la couche OSI où se situe l'équipement :(1 répéteur, 2pont,...)De type Integer (0..127).

2- Le groupe interface (1.3.6.1.2.1.2)

Il contient les informations concernant toutes les interfaces du système administré. Une station de travail n'aura qu'une interface en principe, par contre un routeur en aura au moins deux. Ce groupe défini donc une structure qui sera répétée pour chaque interface que l'équipement possède. Elle est indexée sur le numéro d'interface. La plate forme d'administration peut changer l'état des interfaces pour les activer ou les inhiber.

IfNumber : Le nombre d'interfaces réseaux sur lesquelles ce system peut envoyer/recevoir des datagrammes IP.

La table des inerfaces :

IfTable : Une liste d'entrées d'interfaces. Le nombre d'entrées est donné par la valeur de *ifNmber*.

IfEentry : Une entrée d'interface contenant des objets de couche sous-résseau et dessous pour une interface particulière.

Les composants individuels de chaque entrée interface.

Ifindex : Une valeur unique pour chaque interface. Sa valeur est comprise entre1 et la valeur de *ifnumber*. Cette valeur doit rester constante, du moins de la rè-initialisation du système d'administration rèseau jusqu'à la prochaine ré-initialisation.

IfDescr : Une chaîne textulle contenant une information à propos de l'infrace. Cette chaîne doit inclure le nom de fabrication, le nom du produit et la version de l'interface hardware. La chaîne est prévue pour être présentée à un humain, elle ne doit pas contenir n'importe quoi mais des caractères ASCII imprimables.

IfType : Le type d'interface, distingué selon le protocole réseau/lien/physique immédiatement sous IP dans la pile des protocoles.

IfMtu : La taille en octets du plus large datagramme IP qui peut être envoyer/reçu sur l'interface.

IfSpeed : Une estimation de la largeur de bande courante en bits/sec. Pour des interfaces qui ne varient pas en largeur de bande ou pour celles où aucune estimation juste peut être faite, cet objet doit contenir la largeur de bande nominale.

IfPhysAddress : L'adresse de l'interface au niveau de la couche protocole immédiatement au-dessous de IP dans la pile de protocole. Pour les interfaces qui n'ont pas une telle adresse, cet objet doit contenir une chaîne d'octets de taille 0.

IfAdminStatus : L'état désiré de l'interface. L'état testeur indique qu'aucun paquet opérationnel peut être passé.

IfOperStatus : L'état opérationnel courant de l'interface. L'état testeur indique qu'un paquet opérationnel peut être passé.

IfLastChange : La valeur de '*sysUptime*' au moment où l'interface a entré son état opérationnel courant. Si l'état courant a été entré avant la dernier ré-initialisation du sous-systeme d'administration réseau local alors cet objet contient une valeur 0.

IfInOctets : Le nombre total d'octets reçus sur l'interface incluant les caractères de trame.

IfInUcastPkts : Le nombre de paquets unicast (sous-résseau) délivrés à un protocole de la couche supérieure.

IfInNUcastPKts : C'est le nombre de paquets broadcast reçus.

IfInDiscards : Le nombre de paquets choisi, pour être jetés même bien qu'aucune erreur a été détectée pour prévenir leur délivrance à une couche protocole supérieure. Une raison possible pour jeter un tel paquet pourrait être libérer buffer.

IfInErrors : Le nombre de paquets qui contiennent des erreurs les empêchant d'être délivrés à une couche protocole supérieure.

IfInUnKnownProtos : Le nombre de paquets reçus via l'interface, qui sont jetés à cause d'un protocole nom supporté ou inconnu.

IfOutOctets : Le nombre total d'octets transmis en dehors de l'interface, incluant les caractères de trame (compteur incrément).

IfOutNUcastPkts : Nombre de paquets braodcast et multicast émis (compteur incrément).

IfOutUcastPkts : Le nombre de paquets hors limite qui sont choisis, pour être jetés même bien qu'aucune erreur a été détectée pour prévenir leur délivrance à une couche protocole supérieure. Une raison possible pour jeter un tel paquet pourrait être pour libérer l'espace buffer.

ifOutErrors : Le nombre de paquet or limite qui pourraient être transmis à cause d'erreurs

ifOutQLen : La longueur de la queue des paquets en sortie.
ifSpecifie : Elle fournit une information à propos d'une information spécifique sur le support utilisé pour réaliser l'interface.

3- le groupe adresse translation(1.3.6.1.2.1.3)

Ce groupe contient une table qui est l'union de toutes les interfaces des tables de traduction pour convertir une adresse réseau (ex IP) en une adresse physique.
Exemples de telles tables de traduction : dans le cas des broadcast ou ARP (adresse résolution protocole) est utilise dans la traduction.

AtTable : Les tables de traduction d'adresse contiennent les équivalences entre adresse réseau et adresse physique, certaines interfaces n'utilisent pas de telles tables pour déterminer des équivalences d'adresse (ex : DDN-X.25 a une méthode algorithmique).
AtEntry : Chaque entrée contient une adresse équivalente à une adresse physique.
Nous considérons maintenant les composants individuels de chaque entrée de table de traduction d'adresses.
AtIfIndex : L'interface sur laquelle cette équivalente d'entrée est effective, l'interface est identifiée par une valeur particulière de cet index est la même interface qui est identifiée par la même valeur de *ifIndex*.
AtPhyAddress : l'adresse physique dépendant du support.
AtNetAddress : l'adresse réseau correspondant à l'adresse physique dépendante du support.

4- Le groupe IP (1.3.6.1.2.1.4)

Il contient les informations propres au protocole IP. Il contient trois tableaux, liés à chaque interface :
- Le premier décrit la configuration locale des interfaces pour le protocole IP, chaque interface possède une seule configuration IP, l'indexation sur le numéro de l'interface est suffisante.
- Le second donne les tables de routage, il est indexé sur la destination de la route, ceci peut parfois conduire à des ambiguïtés.
- Le dernier contient les tables de correspondances entre adresse IP et adresse de la couche inférieure. Ce dernier tableau remplace le groupe at, il est indexé sur le numéro de l'interface et sur le type de média.

Les informations en écritures permettent à la plate forme d'activer ou d'inhiber des interfaces et de modifier des informations dans les tables de correspondance.

IpForwarding: Indique si l'équipement sert de routeur(1) ou s'il rejette les paquets qui ne lui sont pas adressées. De type Integer.
IpDefaultTTL: Valeur par défaut de la durée de vie placée dans les paquets à l'émission. De type Integer.
IpInReceives: Nombre de paquets reçus des interfaces. De type Counter.
IpInHdrErrors: Nombre de paquets rejeté à cause d'erreurs dans leur en-tête , incluant les mauvais checksums, erreurs de format time to live dépassé, erreurs découvertes en traitant leurs option IP,ect…De type Counter.
IpInAddrErrors: Nombre de paquets reçus rejetés à cause d'une fausse adresse. A cause d'erreur de l'adresse IP dans leur champs destination, ceci inclus les adresses invalides (ex0.0.0.0) et les adresse de classes non supportées (ex classe E). De type Counter.
IpForwadDatagrams : Nombre de paquets routés (réémis sur une autre interface). De type Counter.

IpInUnknownProtos: Le nombre de datagrammes adressés localement reçus avec succès mais jetés à cause d'un protocole inconnu ou non supporté. De type Counter.
IpInDiscards: : Le nombre d'entrées de datagramme IP pour lesquels aucun problème n'a été rencontré pour empêcher la continuité de leur traitement, mais qui sont jetés (ex : par manque d'espace buffer). Notons que ce compteur n'inclus aucun datagramme jeté pendant l'attente de ré-assemblage. De type Counter.
IpInDelivers: Le nombre total d'entrées de datagramme délivrés avec succès à des protocoles utilisateurs IP (incluant ICMP). De type Counter.
IpOutRequest : Le nombre totale de datagrammes **IP** dont les protocoles utilisateurs IP locaux fournissent à IP dans des requêtes pour transmission(Nombre de paquets soumis par les couches supérieures incluant ICMP). Ce compteur n'inclus aucun datagramme compté dans *ipforwarddatagrams*. De type Counter.
IpOutDiscards: Le nombre de sorties de datagrammes IP pour lesquels aucun problème n'a été rencontré pour empêcher la continuité de leur traitement, mais qui sont jetés(ex : par manque d'espace bufer). Notons que ce compteur inclurait les datagrammes comptés dans *ipforwarddatagrams* si les paquets rencontrent ce critère de suppression. De type Counter.
IpOutNoroutes : Le nombre de datagrammes IP jetés car aucune route n'a pu être trouvée pour les transmettre à leur destination. Notons que ce compteur inclus n'importe quel paquet compté *ipforwardDatagrams* qui rencontre ce critère de (no-route). De type Counter.
IpReasmTimeout: Le nombre maximum de secondes dont les fragments reçus sont tenus pendant qu'ils attendent d'être réassemble à cette entité. De type Counter.
IpReasmReqds: Le nombre de fragments IP reçus qui ont besoin d'être réassemblés à cette entité. De type Counter.
IpReasmOks : Le nombre de datagrammes IP réassemblés avec succès. De type Counter.
IpReasmFails : Le nombre de pannes détectées par l'algorithme de réassamblage. Notons que cela n'est pas nécessairement un compteur de fragments IP jetés puisque certains algorithmes peuvent perdre la piste d'un nombre de fragments en les combinant comme ils ont été reçus. De type Counter.
IpFragOks : Le nombre de datagrammes IP qui ont été fragmentés avec succès à cette entité (compteur incrément).
IpFragFails : Le nombre de datagrammes IP qui ont été jetés parce qu'ils avaient besoin d'être fragmentés à cette entité mais ne pouvaient pas être, par leur flag « dont fragment » (compteur incrément).
IpFragCreates : Le nombre de fragments de datagrammes IP qui ont été généré comme résultat d'une fragmentation. De type Counter.
IpRoutingDescards : Elle fournit une information sur les routes perdues à cause de l'espace buffer. De type Counter.

La table des adresses IP :
IpAddrTable : La table de l'information d'adressage qui a rapport aux adresses IP de cette entité.
IpAddrEntry : L'information d'adressage pour une des adresses IP de cette entité.
IpAdEntIndex : La valeur d'index qui identifie uniquement l'interface pour laquelle cette entrée est applicable. L'interface identifiée par une valeur particulière de cet index est la même interface identifiée par la même valeur de *ifIndex*.
IpAdEntNetMasK : Le masque de sous réseau associé à l'adresse IP de cette entrée. La valeur du masque est une adresse IP avec tous les bits identifiant le réseau sont mis à 1 et tous les bits identifiant l'hôte sont mis à 0.
IpAdEntBcastAddr : La valeur du dernier bit significatif dans l'adresse de broadcast utilisée pour envoyer les datagrammes sur l'interface associée avec l'adresse IP de cette entrée.

IpAdEntReasmMax : Elle garde une trace du plus grand datagramme IP qui peut être réasemblé sur une interface particulière.

La table de routage IP :
Cette table contient une entrée pour chaque route connue de cette entité.
IpRoutingTable : La table de routage de cette entité.
IpRouteEntry : Une route pour une destination particulière.

Les composants individuels de chaque route dans la table de routage :
Ip RouteDest : L'adresse IP destination de cette route. Une entrée avec une valeur 0.0.0.0 est considérée comme route par défaut. De multiples routes par défaut peuvent apparaître dans la table, mais l'accès de tels entrées multiples est dépendant des mécanismes d'accès à la table définis par le protocole d'administration de réseau utilisé.
IpRouteIfIndex : La valeur d'index qui identifie uniquement l'interface locale à travers laquelle la prochaine étape de cette route devrait être atteinte. L'interface identifiée par une valeur particulière de cet index est la même interface identifiée par la même valeur de *ifindex*.
IpRouteMetric1 : Le premier routage metric pour cette route. Les sémantiques de ce metric sont déterminées par le protocole de routage spécifié dans la valeur *iprouteproto* de la route. Si ce metric n'est pas utilisé sa valeur doit être-1.
IpRouteMetric2 : Le premier routage metric pour cette route. Les sémantiques de ce metric sont déterminées par le protocole de routage spécifié dans la valeur *ipRouteProto* de la route. si ce metric n'est pas utilisé sa valeur doit être-1
IpRouteMetric3 : Le premier routage metric alterné pour cette route. Les sémantiques de ce metric sont déterminées par le protocole de routage spécifié dans la valeur *ipRouteProto* de la route. si ce metric n'est pas utilisé sa valeur doit être-1
IpRouteMetric4 : Le premier routage metric alterné pour cette route. Les sémantiques de ce metric sont déterminées par le protocole de routage spécifié dans la valeur *ipRouteProto* de la route. Si ce metric n'est pas utilisé sa valeur doit être-1.
IpRouteMetrix5 : Métrique de routage 5.
IpRouteNextHop : L'adresse IP de la prochaine étape de cette route.
IpRoutetype : Type de routage (directe, distante, valide, invalide).
IpRouteProtol : Le mécanisme de routage via lequel cette route a été apprise. L'inclusion des valeurs pour les protocoles de routage passerelle n'est pas destinée à impliquer que les hôtes doivent supporter ces protocoles.
IpRouteAge : Le nombre de secondes depuis que cette route a été mise à jour ou autrement déterminer pour être correcte. Notons que les sémantiques de 'too old' peuvent être impliquées excepté à travers l'acquittement du protocole de routage par lequel la route a été apprise.
IpRoutMasK : Masque du sous réseau de la route.
IpRouteInfo : Pointeur vers une MIB spécifique de routage.

Table de résolution d'adresse IP :
Contient les correspondances entre les adresses IP et les adresses physiques.

IpNetToMediaEntry : Définit une ligne dans la table *IpNetToMediaTable* : chaque ligne contient quatre colonnes :
IpNetToMediaIfIndex : Le numéro de l'interface.
IpNetToMediaPhysAddress : Adresse physique de la correspondance.
IpNetToMediaNetAdresse : Adresse IP de la correspondance.
IpNetToMediaType : Mode d'établissement de la correspondance.

5- Le groupe ICMP (1.3.6.1.2.1.5)

Il contient les informations propres au protocole ICMP. Il ne contient que des relèves de Compteur (statistique).

IcmpInMsgs : Le nombre total de messages ICMP que l'entité a reçu. Notons que ce compteur inclus tous ceux qui ont comptés par *icmpInErorrs*. De type Counter.
IcmpInErrors : Le nombre de messages ICMP que l'entité a reçu mais déterminés comme ayant des erreurs (mauvais checksum ,mauvaise longueur..). De type Counter.
IcmpInDestUnreachs : Le nombre messages ICMP "destination Unreachable' inaccessible' " reçu. De type Counter.
IcmpInTimeExchs : Le nombre de messages ICMP "Time Exceeded' durée de vie dépassée' "reçu. De type Counter.
IcmpInParmProbs : Le nombre de messages "parameter problem"reçu. De type Counter.
IcmpInSrcQuenchs : Le nombre de messages "Source Quench " reçu. De type Counter.
IcmpInRedirects : Le nombre de messages "Redirect"reçu. De type Counter.
IcmpInEchos : Le nombre de messages "Echo Request" reçu. De type Counter.
IcmpInEchoReps : Le nombre de messages "Echo Reply" reçu. De type Counter.
IcmpInTimestamps : Le nombres de messages "Timestamps Request " reçu. De type Counter.
IcmpInTampReps : Le nombres de messages "Timestamps replyt" reçu. De type Counter.
IcmpInAddrMasks : Le nombre de messages "Address Mask Request" reçu. De type Counter.
IcmpInAddrMaskReps : Le nombre de messages "Address Mask Reply" reçu. De type Counter.
IcmpOutMsgs : Le nombre de messages ICMP que l'entité a tenté d'envoyer.Notons que ce compteur inclut tous ceux qui sont comptés par *icmpInOutErrors*. De type Counter.
IcmpOutError : Le nombre de messages ICMP que l'entité n'a pas envoyé à cause de problèmes découverts à l'intérieur de ICMP tel qu'un manque d'espace buffer. Cette valeur ne doit pas inclure les erreurs découvertes en dehors de la couche ICMP telles que l'incapacité de IP à router le datagramme résultant. De type Counter.
IcmpInOutDestUnreachs : Le nombre de messages ICMP "Destination Unreachable" envouyés. De type Counter.
IcmpInOutTimeExcds : Le nombre de messages ICMP "Time Exceeded" envoyés. De type Counter.
IcmpOutParmProbs : Le nombre de messages ICMP "Parametre Problem" envoyés. De type Counter.
IcmpOutSrcQuenchs : Le nombre de messages ICMP "Source Quen c h" envoyés. De type Counter.
IcmpOutRedirects : Le nombre de message ICMP "Redirect"envoyés. De type Counter.
IcmpOutEchos : Le nombre de message ICMP "Echo Request" envoyés. De type Counter.
IcmpOutEchoReps : Le nombre de message ICMP "Echo Reply" envoyés. De type Counter.
IcmpOutTimestamps : Le nombre de messages ICMP "Timestamp Request" envoyés. De type Counter.
IcmpOutTimestampReps : Le nombre de messages ICMP "timestamp Reply" envoyés. De type Counter.
IcmpOutAddrMasks : Le nombre de messages ICMP "adress Mask Request" envoyés. De type Counter.

IcmpOutAddrMaskReps : Le nombre de messages ICMP "Adress Mask Reply"envoyés. De type Counter.

6- Le groupe TCP(1.3.6.1.2.1.6)

Il contient les informations propres au protocole TCP. Il possède un tableau ayant une entrée par connexion. L'indexation se fait sur les champs adresse source, adresse destination, port source et port destination, qui identifient de manière unique une connexion.

TcpRtoalgorithm : L'algorithme utilisé pour déterminer la valeur de timeout pour retransmettre des octets acquittés. De type Integer.
TcpRtoMin : La valeur minimum permise par une implémentation TCP pour le timeout de retransmission, mesure en milliseconde. De type Integer.
TcpRtoMax : La valeur maximum permise par une implémentation TCP pour le timeout de retransmission, mesuré en milliseconde. De type Integer.
TcpMaxConn : Nombre maximum de connexions simultanées (-1 pas de limite). De type Integer.
TcpActiveOpens : Le nombre de fois que des connexions TCP ont fait une transition directe à l'état SYN-RCVD de l'état LISTEN (Nombre de connexions ouvertes à l'initiative de l'équipement). De type Counter.
TcpPassiveOpens : Nombre d'ouvertures de connexion passives(Nombre de connexions ouvertes à l'initiative de l'équipement distant). De type Counter.
TcpAttemtFails : Nombre de tentatives d'ouvertures de connexion qui ont échoué. De type Counter.
TcpEstabResets : Nombre de connexions qui ont été fermées par un paquet RESET. De type Counter.
TcpCurrEstab : Nombre actuel de connexions ouvertes. Le nombre de connexion TCP pour lesquelles l'état courant est soit Estabshed, soit Close-Wait. De type Counter.
TcpInSegs : Nombre total de segments reçus (incluant ceux en erreur). De type Counter.
TcpOutSegs : Nombre de segments émis (sans compter les retransmission). De type Counter.
TcpRetransSegs : Le nombre total de segments retransmis. De type Counter.
TcpInErrs : Nombre de segments ignorés à cause d'une erreur de format.
TcpOutRsts : Nombre de Reset générés.

Table de connexion TCP :
TcpConnTable : Une table contenant une information de connexion TCP spécifique. De type TcpConnEntry :13.1.
TcpConnEntry : Information à propos d'une connexion TCP courante particulière. Un objet de ce type est éphémère, il cesse d'exister quand la connexion fait la transition à l'état CLOSED, chaque ligne de la table comporte cinq colonnes :

TcpCnnState : L'état de cette connexion TCP.
TcpConnLocalAdress : Adresse IP locale pour la connexion. 0.0.0.0 indique que la station accepte des connexions de n'importe quel équipement. De type IpAdress.
TcpConnLocalPort : Le numéro du port local pour cette connexion TCP. De type Integer (0..65535).
TcpConnRemAddress : L'adresse IP distante pour cette connexion TCP. De type IpAdress.
TcpConnRemPort : Numéro de port distant. De type Integer (0..65535).

7- Le groupe UDP (1.3.6.1.2.1.7)

Il contient les informations propres au protocole UDP. Il possède un tableau contenant les valeurs locales des sockets.
UdpInDatagrams : Le nombre total de datagrammes UDP délivrés par les utilisateurs UDP.
UdpNoPorts : Le nombre total de datagrammes UDP reçus pour lesquels il n'y avait pas d'application au port destination.
UdpInError : Le nombre de datagrammes reçus qui ne peuvent pas être délivrés pour des raisons autres que le manque d'application au port destination.
UdpOuDatagrams : Le nombre total de datagrammes UDP envoyés de cette entité.

La table udp Table :
UdpLocalAddress : Adresse IP locale.
UdpLocalPort : Port UDP locale.

8- Le groupe EGP (1.3.6.1.2.1.8)

Il contient les informations propres aux protocoles EGP. L'implémentation de ce groupe est obligatoire pour tous les systèmes qui implémentent le protocole EGP.

EgpInMsgs : Le nombre total des messages EGP reçus sans erreur.
EgpInErrors : Le nombre total des messages EGP reçu prouvés en erreur.
EgpOutMsgs : Le total de messages EGP localement produits.
EgpErrors : Le nombre de messages EGP générés localement non envoyés à cause des limitations de ressources à l'intérieur d'une entité EGP.

9- Le groupe transmission (1.3.6.1.2.1.10)

Il contient les objets relatifs aux mécanismes de transmission. La MIB SNMP ne contient pas d'objets sous cet OID. Les objets transmissions sont définis dans d'autres RFC.

10- Le groupe SNMP (1.3.6.1.2.1.11)

Ce groupe contient les objets relatifs à la version V1 de SNMP. Il comporte 30 variables
SnmpInPkts : Nombre total de PDU reçues de la couche inférieure. De type Counter.
SnmpOutPkts : Nombre total de PDU envoyées à la couche inférieure. De type Counter.
SnmpBadVersions : Nombre total de PDU reçues ayant un numéro de version différent du numéro local. De type Counter.
SnmpBadCommunityNames : Nombre total de PDU ayant un nom de communauté inconnu. De type Counter.
SnmpBadCommunityUses : Nombre total de PDU ne pouvant être traités dans le cadre de cette communauté. De type Counter.
SnmpInASNParseErrs : Nombre total d'erreurs au moment de l'interprétation d'un objet ANS.1 (BER Basic Encoding Rule : codage ou syntaxe).
SnmpInBadTypes : Nombre de PDU avec un type inconnu.

Code de retour :
SnmpInTooBigs : Nombre de PDU reçues avec comme ErrorStatus'too bigs'(la réponse ne tient pas dans un message UDP). De type Counter.
SnmpInNoSuchNames : Nombre de PDU reçues avec comme ErrorStatus 'noSuchNames' (nom inconnu). De type Counter.

SnmpInBadValues : Nombre de PDU reçues avec comme ErrorStatus 'badValue'. De type Counter.
SnmpInReadOnlys : Nombre de PDU reçue avec comme ErrorStatus'readOnly'. De type Counter.
SnmpInGenErrs : Nombre de PDU reçue avec comme ErrorStatu 'genErr'. De type Counter.

Objets Statiques en Entrée :
SnmpInTotalReqVars : Nombre d'objets de la MIB qui ont fait l'objet d'une requête Get-Request ou get-Next.
SnmpInTotalSetVars : Nombre d'objets de la MIB qui ont fait l'objet d'une requête Set-Request.
SnmpInGetRequests : Nombre total de PDU Get-Request traitées par le protocole SNMP. De type Counter.
SnmpInGetNexts : Nombre total de PDU Get-Nex traitées par le protocole SNMP. De type Counter.
SnmpInSetRequests : Nombre total de PDU Set-Request traitées par le protocole SNMP.
SnmpInGetRsponses : Nombre total de PDU Get-Rsponse traitées par le protocole SNMP. De type Counter.
SnmpInTraps : Nombre total de PDU Trap traitées par le protocole SNMP. De type Counter.
SnmpOutTooBigs : Nombre total de PDU SNMP qui ont été générés par l'entité de protocole SNMP et pour lesquelles la valeur du champ 'ErrorStatus' est'tooBig'. De type Counter.
SnmpOutNoSuchNames : Nombre total de PDU SNMP qui ont été générés par l'entité de protocole SNMP et pour lesquelles la valeur du champ 'ErrorStatus' est 'noSuchName'. De type Counter.
SnmpOutBadValues : Nombre total de PDU SNMP qui ont été générés pour l'entité de protocole SNMP et pour lesquelles la valeur du champ 'ErrorStatus'est 'BadValue'.
SnmpOutGenErrs : Nombre total de PDU SNMP qui ont été générés par l'entité de protocole SNMP et pour lesquelles la valeur du champ 'ErrorStatus' est 'genErr'. De type Counter.

Objets Statique en Sortie :
SnmpOutGetRequests : Nombre total de PDU de type Get-Request qui ont été générés par l'entité de protocole SNMP. De type Counter.
SnmpOutGetNexs : Nombre total de PDU de type Get-Next qui ont été générés par l'entité de protocole SNMP. De type Counter.
SnmpOutSetRequests : Nombre total de PDU de type Set-Request qui ont été générés par l'entité de protocole SNMP. De type Counter. .
SnmpOutGetResponses : Nombre total de PDU de type Get-Responses qui ont été générés par l'entité de protocole SNMP. De type Counter.
SnmpOutTraps : Nombre total de PDU de type Trap qui ont été générés par l'entité de protocole SNMP. De type Counter.
SnmpEnableAuthTraps : Indication sur la configuration de l'agent SNMP concernant la génération du Trap d'Authentification. De type INTEGER.

II- La MIB du modèle USM définit pour snmpv3

La MIB SNMP-USER-BASED-SM-MIB contient les informations concernant le modèle de sécurité USM (User Based Security Model) ainsi que les protocoles de sécurité utilisés.
Dans ce qui suit nous allons définir les objets et les tables constituant cette MIB.

1-Les identificateurs des protocoles d'authentification et de cryptage

UsmNoAuthProtocol : Aucun protocole d'authentification n'est appliqué.
UsmHMACMD5AuthProtocol : Le protocole d'authentification HMAC-MD5-96 est appliqué.
UsmHMACSHAAuthProtocole : Le protocole d'authentification HMAC-SHA-96 est appliqué.
UsmNoPrivProtocol : Aucun protocole de cryptage n'est appliqué.
UsmDESPrivProtocol : Le protocole de cryptage DES est appliqué.

1-1- les statistiques de USM-Model

UsmStatsUnsupportedSecLevels : Le nombre total de paquets reçus et ignorés par un moteur SNMP, à cause de leur niveau de sécurité inconnu.
UsmStatsUnknownUserNames : Le nombre total de paquets reçus et ignorés par un moteur SNMP, car l'uitlisateur mentionné est inconnu.
UsmStatsUnknownEngineIDs : Le nombre total de paquets reçus et ignorés par un moteur SNMP, car ils font référence à un moteur snmpEngineID inconnu.
UsmStatsWrongDigests : Le nombre total de paquets reçus et ignorés par un moteur SNMP, puisque la valeur de Digest (résultat de l'algorithme de hachage) n'est pas acceptée
UsmStatsDescryptionErrors : Le nombre total de paquets reçus et ignorés par un moteur SNMP, parce qu'il n'a pas pu les décrypter.

1-2- le groupe des utilisateurs

UsmUserSpinLock : Un signal de fermeture est utilisé pour permettre à plusieurs applications Command Genetator de se coordonner, pour faciliter le changement de mot de passe secret qui se trouve dans la table UsmUserTable.

La table des utilisateurs de User Based Security Model :
UsmUserTable : La table des utilisateurs configurée dans le moteur SNMP
UsmUserEntry : Une entrée dans la table, définit le profile des utilisateurs configurés dans le moteur SNMP. Chaque entrée est indexée par :

UsmUserEngineID : Identificateur administratif (qui est unique) du moteur SNMP. Dans un agent simple, cette valeur est toujours la propre valeur de snmpEngineID de cet agent. Cette valeur peut également prendre la valeur de snmpEngineID d'un moteur à distance avec lequel cet utilisateur peut communiquer.
UsmUserName : Une chaîne de caractère qui représente le nom d'utilisateur et qui dépend du modèle de sécurité User Based Security Model.
UsmUserSecurityName :Une chaîne de caractère qui représente le nom d'utilisateur dans un format indépendamment du modèle de sécurité. Par défaut securityName et userName sont identiques.
UsmUserCloneFrom : Un pointeur vers d'autres entrées de la table UsmUserTable. Les utilisateurs des autres entrées sont appelés CloneFromuser.
UsmUserAuthProtocol : Indique si les messages envoyés au nom de l'utilisateur UsmUserName par un moteur identifié par UsmUserEngineID, peuvent être authentifiés. Si oui, il indique le protocole d'authentification employé.
UsmUserPublic :Une valeur publique qui peut être écrite en tant qu'éléments de la procédure pour changer la clé secrète d'authentification (ou de cryptage) d'un utilisateur [RFC 1274].

III- La MIB du modèle de contrôle d'accès VACM

Les standards snmpv3 définissent View-Based-Access-Model (VACM), un modèle offrant un ensemble de services aux applications pour vérifier les droits d'accès aux informations de gestion.

1- vacmContxtTable

C'est la table des contextes (Context) disponibles, elle fournit les informations à l'application Command Generator pour qu'elle puisse configurer correctement vacmAccessTable qui contrôle l'accès de tous les contextes de l'entité SNMP.
vacmContxtEntry : Une entrée de la table contenant des informations d'un Context particulier.
vacmContxtName : Un nom lisible qui identifie un Context particulier d'une entité particulière. Par défaut, ContextName est la chaîne de caractère vide.

2- Information de groupe

VacmSecurityToGroupTable : Cette table contient la combinaison de <SecurityModel, SecurityName> dans un GroupName. Elle est utilisée pour définir les politiques de contrôle d'accès d'un groupe d'utilisateurs.
VacmSecurityToGroupEntry : C'est une entrée de la table vacmSecurityToGroupTable, elle est indexée par < vacmSecurityModel, vacmSecurityName>.
VacmSecurityModel : Le modèle de sécurité SecurityModel appliqué. Cet objet ne peut pas être vide.
VacmSecurityName : Le nom SecurityName représentant l'utilisateur dans un format indépendant du modèle de sécurité.
VacmGroupName : Le nom d'un groupe auquel la combinaison <Securityodel, SecurityName> appartient. Le GroupName est utilisé comme index dans la table vacmAccessTable pour sélectionner la politique de contrôle d'accès.
VacmSecurityToGroupStatus : Le statut (status) de cette entrée conceptuelle. Si les valeurs de toutes les colonnes d'une entrée sont convenablement configurés, la valeur de la colonne vacmSecurityToGroupStatus est « notReady ». En particulier, une entrée nouvellement créée, ne peut pas être rendue active jusqu'à ce qu'une valeur soit attribuée pour vacmGroupName.

3- Informations de Access Control

VacmAccessTable : Une table des droits d'accès des groupes. Chaque entrée est indexée par <GroupName, ContextPrefix, SecurityModel, SecurityLevel >. Pour déterminer si l'accès est permis, une entrée de cette table doit être sélectionner et le viewName approprié doit être employé pour la vérification de contrôle d'accès.
VacmAccessEntry : Chaque entrée de la table vacmAccessTable est caractérisée par :
VacmAccessContextPrefix : Pour avoir le droit d'accès à un objet, le contexteName doit ressemblé exactement (si la valeur d'instance d'objet vacmAccessContextMatch est exacte) ou partiellement (si la valeur d'instance d'objet vacmAccessContextMatch est prefix) à la valeur d'instance de cet objet.
VacmAccessSecurityModel :Le SecurityModel est utilisé aussi pour vérifier les droit d'accès

VacmAccessSecurityLevel : Le niveau minimum de sécurité exigé pour avoir un droit d'accès de cette entrée. Le niveau noAuthNoPriv est moins que le niveau authNoPriv qui est à son tour moins que authPriv.
VacmAccessContextMatch : Si la valeur de cet objet est exacte, alors toutes les entrées où le ContextName ressemble exactement à vacmAccessContextPrefix sont choisies sinon (la valeur de cet objet est prefix) toutes les entrées où les premiers octets de contextName ressemble à vacmAccessContextPrefix sont choisies.
VacmAccessReadViewName : la valeur d'instance de cet objet identifie la MIB view d'un context auquel l'entrée conceptuelle autorise l'accès en lecture. Cette valeur est la même que celle du vacmViewTreeFamilyViewName. Si cette dernière est vide ou aucune MIB view ayant la valeur de vacmViewTreeFamilyViewName n'est activée, alors l'accès n'est pas permis en lecture.
VacmAccessWriteViewName : La valeur d'instance de cet objet identifie la MIB view d'un context auquel l'entrée conceptuelle autorise l'accès en écriture. Cette valeur est la même que celle du vacmViewTreeFamilyViewName. Si cette dernière est vide ou aucune MIB view ayant la valeur de vacmViewTreeFamilyViewName n'est activée, alors l'accès n'est pas permis en écriture.
VacmAccessNotifyViewName : La valeur d'instance de cet objet identifie la MIB view selon un context auquel l'entrée conceptuelle autorise l'accès en notifications. Cette valeur est la même que celle du vacmViewTreeFamilyViewName. Si cette dernière est vide ou aucune MIB view ayant la valeur de vacmViewTreeFamilyViewName n'est activée, alors l'accès en notification n'est pas permis.
VacmAccessStatus : Le statut (status) de cette entrée.

4- Les informations de la MIB views

VacmViewTreeFamilyTable : Contient les informations sur les familles des sous arbres des MIB view.
Chaque MIB views est définit par deux ensembles de sous arbres de vues : included view subtrees et excluded view subtrees.
VacmViewTreeFamilyEntry : Information d'une famille particulière de sous arbres de vues d'un context particulier qui sont inclus (included) dans la MIB view ou exclus (excluded) de la MIB view. Une entrée de la table vacmViewTreeFamilyTable est indexée par :
VacmViewTreeFamilyViewName : Le nom lisible d'une famille de sous arbre de vues.
VacmViewTreeFamilySubtree : Le sous arbre de la MIB qui, une fois combiné avec la valeur d'instance d'objet vacmViewTreeFamilyMask, on définit une famille de sous arbres de vues.
VacmViewTreeFamilyMask : Un masque qui, une fois combiné avec l'instance d'objet vacmViewTreeFamilySubtree on définit la famille de sous arbres de vues. Si un bit de ce masque est égal à 1, cela signifie que le sous identificateurs d'instance d'objets vacmViewTreeFamilySubtree doit être identique à celui de l'objet qu'on veut avoir l'accès.
VacmViewTreeFamilyType : Indique si les valeurs d'instances d'objets vacmViewTreeFamilySubtree et vacmViewTreeFamilyMask qui définissent une famille de sous arbres de vues, sont inclus (ou exclus) dans la MIB view.
VacmViewTreeFamilyStatus : Le statut (status) de cette entrée.

Bibliographie

Bibliographie

Livres :

[Bru 01] Bruno Bouzy
Cycle de vie de logiciel
Edition Eyrolles 2001

[Joe, Dan 97] Joe Casad Dan newlan
Preparation au MCSE Networking Essentials
Edition Simon & Schuster Mancmillan 1997

[Loa,Véq,Znt 95] Kim Loan, Véronique Véque, Simon Znaty
Arcitecture des réseaux haut bébit
Edition Hermès 1995

[Puj 98] Gue Pujolle
Les réseaux, deuxième édition
Edition Eyrolles 1998

[Puj 00] Gue Pujolle
Les réseaux, troisième édition
Edition Eyrolles 2000

[Ric 94] Richard Stoeckel
Réseaux Communication et Unix
Edition Armand Colin 1994

[Tan 91] Andrew Tanenbaum
Les réseaux troisième édition
Edition Dunod 1991

[Tau 97] Laurent Tautin
Réseaux locaux et Internet
Deuxième édition 1997

Internet :

[Ane 99] Pascal Anelli
QoS au niveau IP : présentation
http://www-rp.lip6.fr/airs/projet1/qos

[Lau 99] Laurent Duchien
L'administration de l'Internet : SNMP 1996
http://tulipe.cnam.fr/personne/duchien/poly.html

[Loz 00] André Lozes
Qos au niveau applicatif 2000
http://www.laas.fr/AIRS/SP2/QoS_appli.html

[Tim 03] Timur Friedman
Qualité de service 2003

[Yli 88] Ylian Saint-Hilaire
SNMPv3-Modulaire : une méthodologie de conception et de mise en œuvre d'un protocole de gestion de réseau 1988
http://quasiturbine.promci.qc.ca/FQTYlianMaitrise.pdf

[RFC 2571] D.Harrington, R.Presuhn, B.Wijnen
An Architecture for Describing SNMP Management Frameworks 1999

[RFC 2574] U.Blumenthal, B.Wijnen
User-based Security Model (USM) for version 3 of the Simple Network Management Protocol 1999

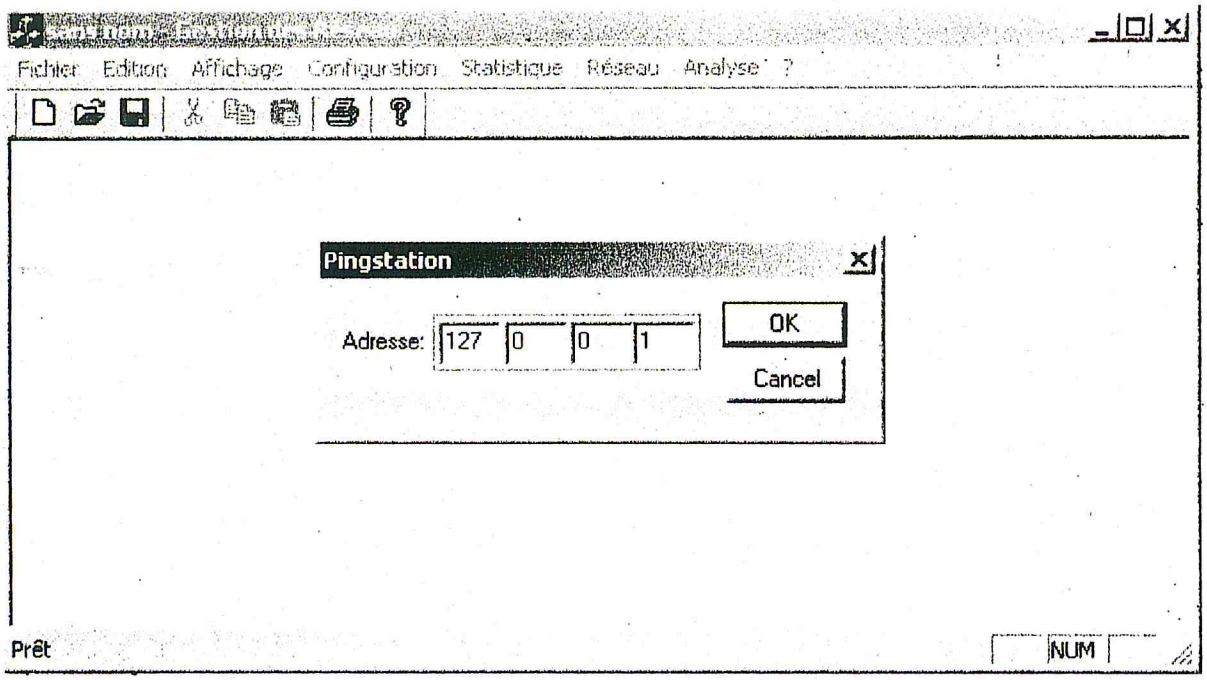


Figure 15 : Ping d'une station

Le Pingstation se fait sur une station particulière.

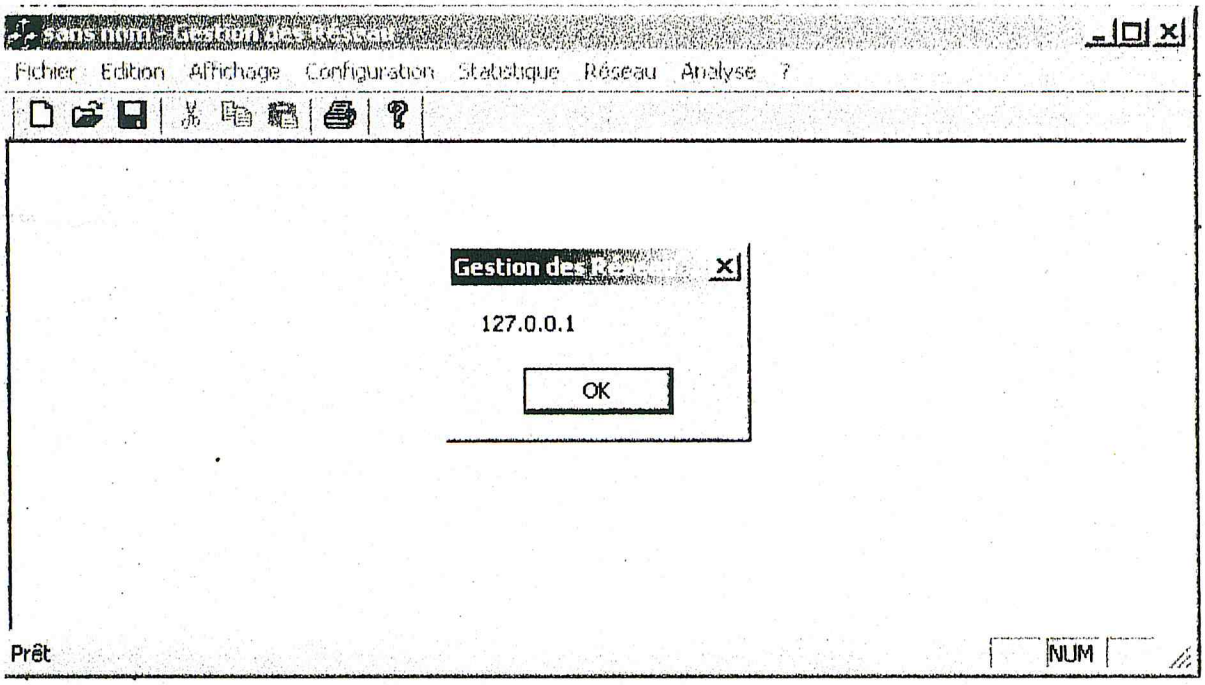


Figure 16 : Résultat du ping

Après avoir lancer le ping dans les deux cas, une boite de dialogue s'affiche pour chaque station active, avec son adresse.

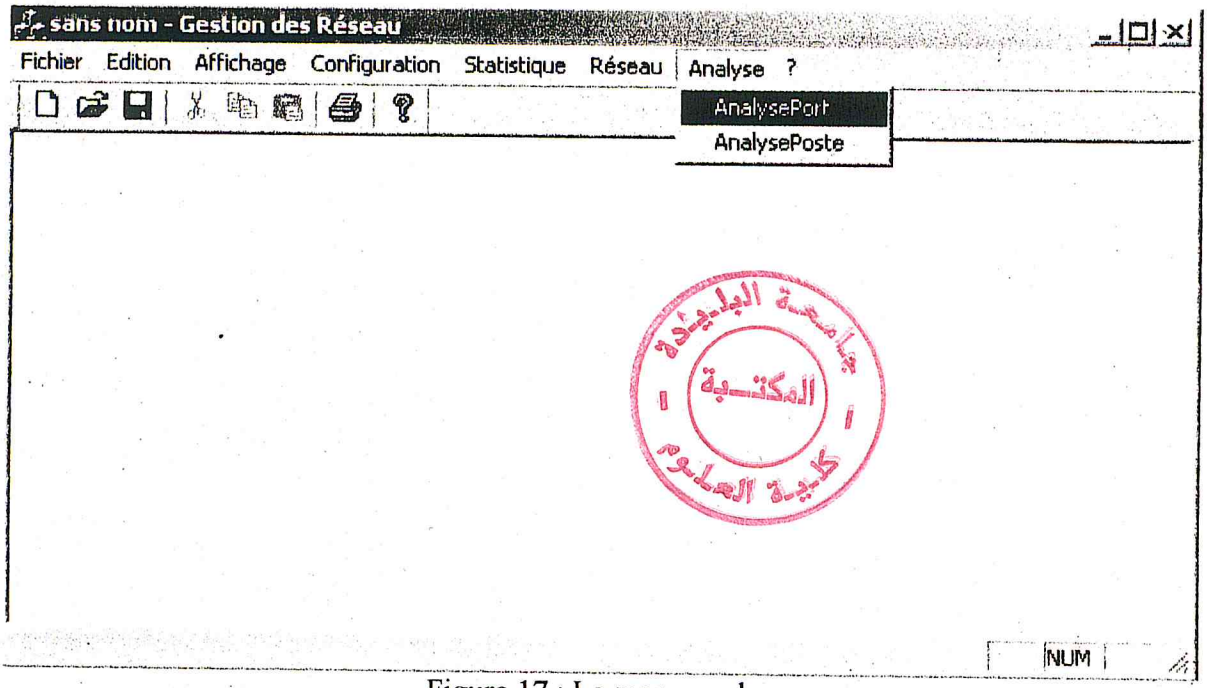


Figure 17 : Le menu analyse

Cette figure montre le menu Analyse qui permet de donner l'analyse par port (par application) et l'analyse par poste (par adresse).

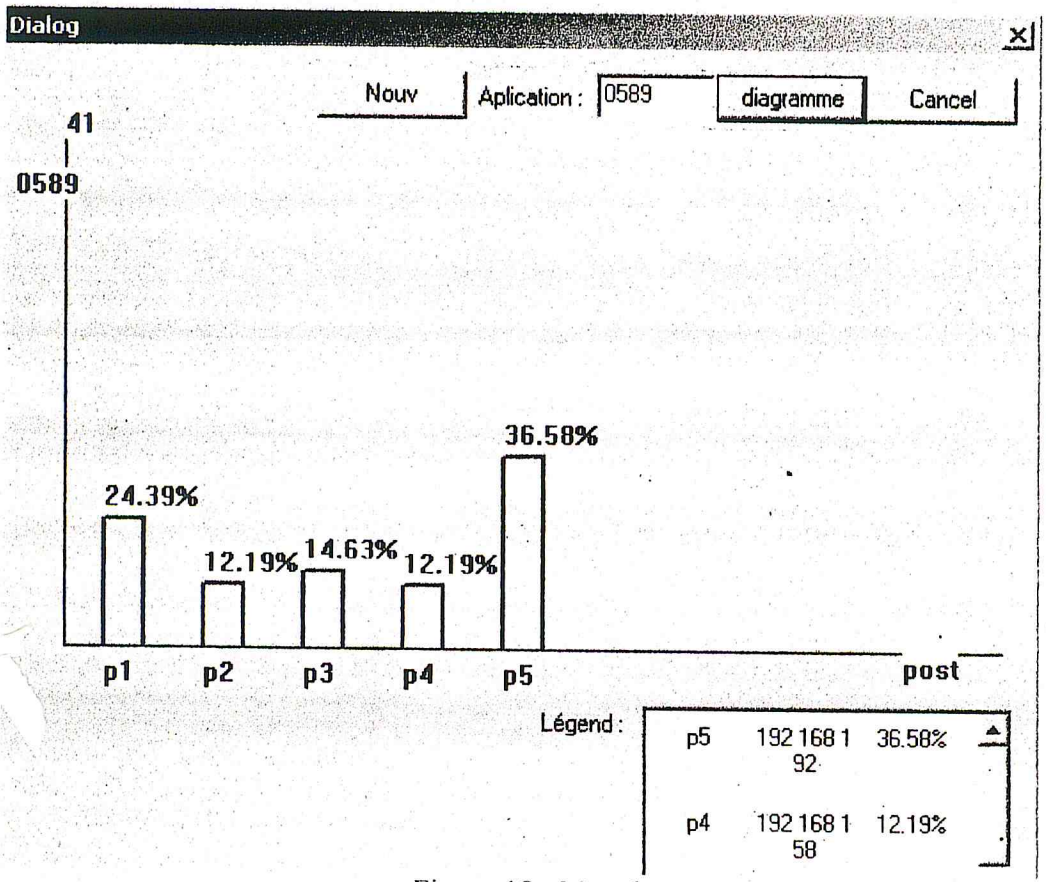


Figure 18 : L'analyse par port

Cette figure montre l'analyse par port (par application), qui consiste à faire une analyse sur une application particulière pour tous les postes.

L'utilisateur fait entré le numéro de port ou bien le nom de l'application et puis il clique sur diagramme pour obtenir le diagramme associé.
New permet de lancer une autre analyse avec une application nouvelle.

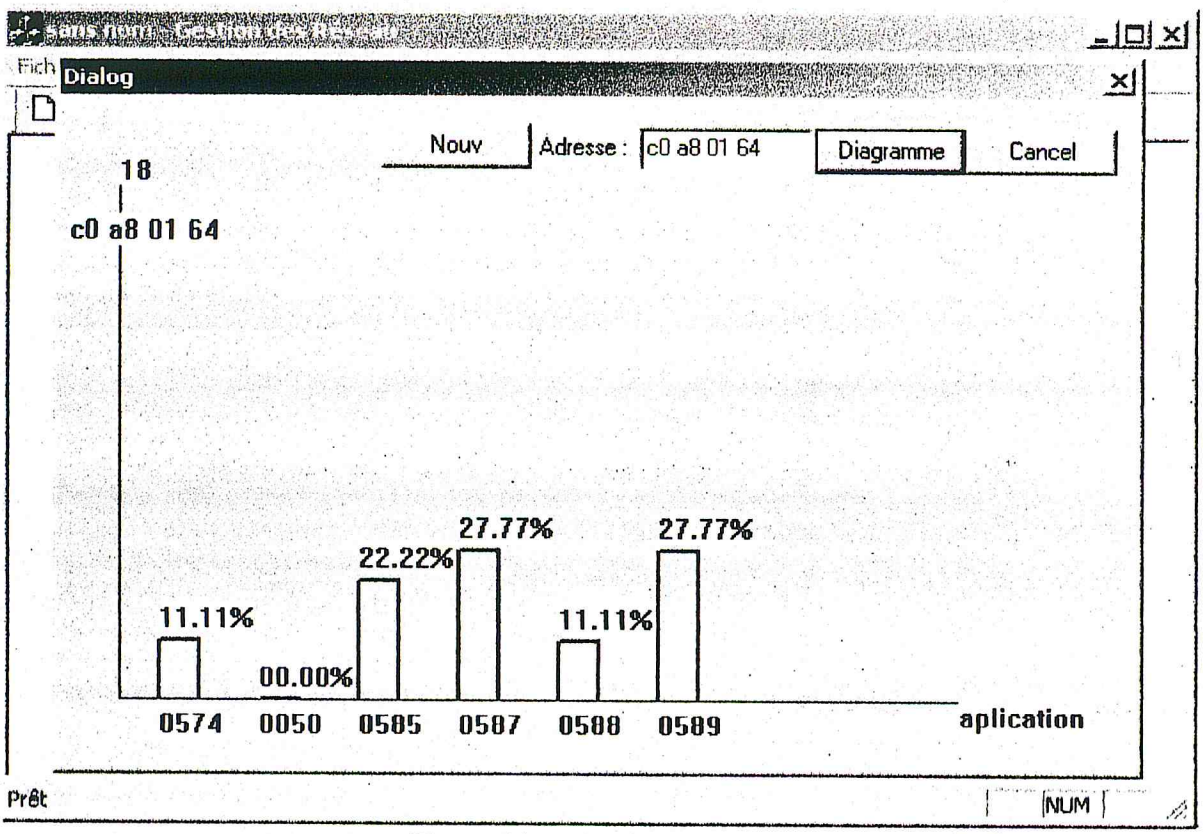


Figure 19 : L'analyse par poste

L'analyse par poste, consiste à faire une analyse sur un poste particulier et voir tous les protocoles utilisés par ce poste.
L'utilisateur fait entrée l'adresse du poste et pour avoir le diagramme, il clique sur le bouton diagrramme. Une analyse d'un autre poste est possible par le bouton new.

IV- Validation

La validation doit être envisagée lors de l'achèvement du travail de développement, une fois que la qualité technique du logiciel est démontrée. Elle permettra de garantir la logique et la complétude du système.
Nous avons choisi de faire la validation par test, le test précédent que nous avons fait, montre que les cas d'utilisation proposés dans le chapitre précédent sont validés.

V Conclusion

L'application a été réalisée avec l'idée d'offrir à l'utilisateur tous les services de notre système, sous une forme agréable et conviviale.
Indiquant tout de même que sa réalisation a été simplifiée par les remarquable richesse et fonctionnalités des composants visuels proposés par l'environnement de développement utilisé.

Conclusion générale

Conclusion générale

Nous avons présenté à travers notre travail, une étude visant à contribuer à la conception et la réalisation d'un logiciel qui réalise la cartographie du réseau en utilisant le protocole SNMP, la capture et l'analyse des paquets transitant sur le réseau, l'extraction des informations à partir des statistiques obtenues afin de déceler les goulots d'étranglement.

Tout au long de la conception de notre projet, nous avons pu affranchir l'agréable monde des réseaux informatiques en général et les réseaux locaux en particulier. Ce monde qui a évolué ces dernières décennies d'une manière fascinante, en permettant de relier des centaines de millions d'ordinateurs distribués partout dans le monde.
L'étude de la pile de protocoles TCP/IP nous a permis de comprendre des notions très importantes concernant les mécanismes de fonctionnement de grands réseaux mondiaux tels que l'Internet, ainsi que les réseaux locaux tels que l'Ethernet. Ces notions nous ont offert une vision plus précis e sur les protocoles et les applications qui utilisent l'Internet pour réaliser leurs différents objectifs.
Ce travail nous a permis de nous actualiser notamment dans le domaine d'administration des réseaux informatiques, mais surtout sur les différentes versions de SNMP.

Enfin, permettez-nous d'affirmer que l'expérience que nous avons vécue dans le cadre de notre mémoire de fin d'étude a été fructueuse. Elle consiste une solide base de départ qui nous a permis d'aborder des travaux plus complexes et d'apprendre en même temps à penser et à comprendre comment exploiter les moyens et les outils disponibles afin de réaliser un travail qui répond au mieux à nos besoins.

En guise de perspectives et au vu des résultats obtenus nous suggérons :

- L'implémentation d'une nouvelle version de SNMP (SNMPv3) qui semble bien adaptée à des problèmes de gestion des réseaux, surtout pour l'aspect sécuritaire.

- L'implémentation de la qualité de service à partir des analyses obtenues de la capture des paquets transitant sur le réseau.

Annexe A

Cette annexe, est une annexe de généralités, elle présentera la définition des réseaux ainsi que les principaux composants qui la constituent.

I- Introduction

Parallèlement aux développements dans les différents domaines, l'informatique a marquée sa présence en évoluant. Par le passé un ordinateur unique satisfaisait à tous les besoins de traitement de l'organisation avec des systèmes informatiques très centralisés. Avec le rapprochement de l'informatique et des télécommunications ; un ensemble d'ordinateurs et de périphériques séparés mais interconnectés de façon directe (par câble) ou indirecte (par modem), qu'on appelle réseau d'ordinateurs, permet un traitement meilleur de l'organisation.
Les différents éléments du réseau communiquent les uns avec les autres grâce à un ensemble prédéfinie de règles appelé protocole.
La disposition des réseaux est appelée topologie du réseau.

1- Objectifs des réseaux

- Le partage des ressources : rend accessible à chaque élément de réseau, les données, les programmes, et les équipements.
- Assurer une meilleure fiabilité.
- Réduction des coûts : meilleur rapport prix/performances par rapport aux grands ordinateurs.
- Augmentation des performances du système lorsque la charge de travail croit, par l'adjonction de processeurs.
- Une autre raison d'installer un réseau d'ordinateurs est de fournir un média de communication puissant entres des personnes séparées par de longues distances.

La section suivante portera sur la définition des sous réseaux et des systèmes autonomes suivis de la section 3 qui décrira les différents types de réseaux, ensuite dans la section 4, nous parlerons des différentes topologies des réseaux.
La section 5 parlera des raccordements physiques, et des équipements téléinformatiques existants.
Dans la section 6, nous présenterons le mode avec connexion et le mode sans connexion.
La section 7 définira l'adressage logique et physique.
Le système en couche intérêt et concept est présenté en section 8 suivi d'une conclusion.

II- Définitions générales

1- Définition des sous réseaux

Le réseau est souvent compris dans le sens réseau de communication. Dans la plupart des grands réseaux, le sous-réseau comporte deux parties distinctes : les lignes de transmission et les éléments de commutation.
Les lignes de transmission (appelées également circuits, canaux ou jonctions) échangent les bits entre machines.
Les éléments de commutation sont des ordinateurs spécialisés dans la connexion d'au moins deux lignes de transmission. Quand des données arrivent sur une ligne entrante, le commutateur« processeur d'interface de messages(Interface Message Processor)»doit sélectionner une ligne de sortie pour les y expédier. Chaque hôte est connecté à un ou plusieurs IMP.
Exemple : Pour le sens de sous réseau, le réseau téléphonique est composé de commutateurs de circuits reliés entre eux par des liaisons à haut débit et aux utilisateurs par des liaisons à

- **Le mode de diffusion** (Interconnexion par un support de transmissions partagé) :
Il consiste à partager un seul support de transmission. Chaque message envoyé par un équipement (ETTD) sur le réseau est reçu par tous les autres. C'est l'adresse spécifique placée dans le message qui permettra à chaque équipement de déterminer si le message lui est adressé ou non.
- **Le mode point à point** (Interconnexion par des supports de transmissions séparés) :
Dans ce mode de diffusion le support physique (le câble) relie une paire d'équipements seulement (séparés). Quand deux éléments non directement connectés entre eux veulent communiquer, ils le font par l'intermédiaire des autres nœuds du réseau.

2- Les différentes topologies physiques

2-1- Topologie en bus

Un réseau en bus relie ses composants (ETTD) par un même câble et fonctionne en mode de diffusion. Le câble coaxial sert typiquement à faire ce type de réseaux.
Cette topologie est utilisée dans les réseaux Ethernet 10 Base 2 et 10 Base 5.
Une seule station peut émettre à la fois, pour cela il y a deux techniques :
1 - Le polling-selecting qui consiste à donner provisoirement le statut de maître à chaque ETTD suivant un ordre déterminé en l'invitant à émettre ensuite à recevoir.
2 - La diffusion consiste à adresser chaque ETTD pour l'identifier. Un ETTD qui reconnaît son adresse prend en considération l'information qui circule.
En bout de bus, un « bouchon » permet de supprimer définitivement les informations pour qu'une autre station puisse émettre.

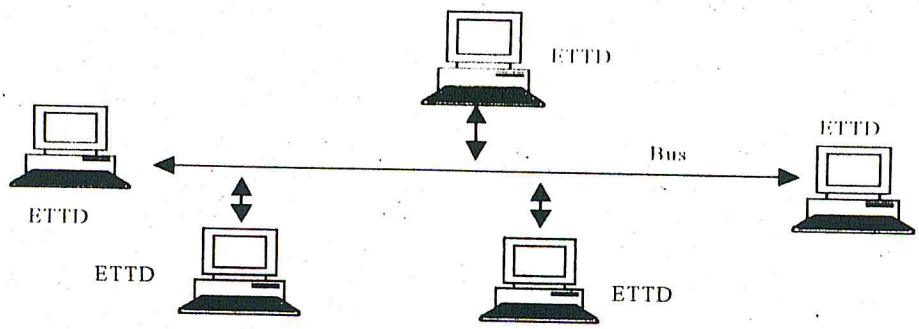


Figure 2 : Topologie en bus

Les avantages de la topologie :
-Si une station tombe en panne cela ne perturbe pas le reste du réseau;
-L'économie du câble;
-Une mise en oeuvre simple et fiable;
-Facile à étendre.
Les inconvénients :
-Ralentissement du trafic en cas de nombreuses stations
-Problème (panne) difficile à isoler.
-Une coupure du câble affecte de nombreux utilisateurs.

2-2- Topologie en anneau

Dans un réseau en anneau, on a une boucle d'ordinateurs (ETTD) sur laquelle chacun d'entre eux va "avoir la parole" successivement. Un jeton circule autour de l'anneau. La station qui a

bas débit. Les lignes et les commutateurs que l'opérateur de réseau possède et qu'il administre forment le sous réseau du réseau téléphonique, les téléphones ici se sont les hôtes qui ne font pas partie du sous réseau, c'est l'ensemble de sous réseau et des hôtes qui forment le réseau. Dans le cas d'un LAN, le support et les hôtes forment le réseau et il n'y a pas de sous réseau.
Remarque : la différence entre un sous réseau et un WAN concerne la présence d'hôte.
S'il y a que des routeurs c'est un sous réseau, mais s'il y a aussi des hôtes avec leur utilisateurs c'est un WAN.

2- Définition d'un système autonome

Un système autonome est défini par un ensemble de routeurs et de réseaux sous une administration unique. Cela peut donc aller d'un seul routeur connectant un réseau local à Internet, jusqu'à l'ensemble des réseaux locaux d'une multinationale. La règle de base étant qu'un système autonome assure la connexité totale de tous les points qui le composent en utilisant notamment un protocole de routage unique IGP et pour obtenir des informations sur les réseaux externes, ils doivent dialoguer avec les routeurs externes grâce à *EGP* (Exterior Gateway Protocol) ou BGP (Border Gateway Protocol) qui remplace EGP actuellement.

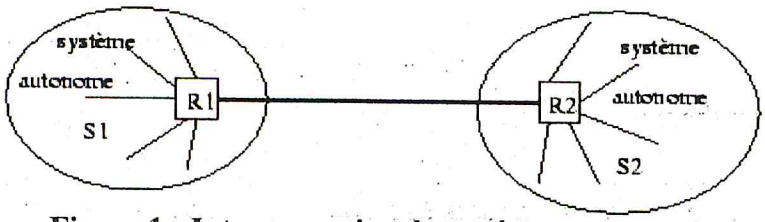


Figure 1 : Interconnexion de systèmes autonomes

III- Type de réseaux

Suivant la distance qui sépare les ordinateurs, on distingue plusieurs catégories de réseaux :
1- Les réseaux locaux LAN (Local Area Network) : qui correspondent par leur taille à des réseaux intro-entreprises. La distance de câblage est de quelques centaines de mètres. Ces réseaux ont des topologies particulières, un type de câblage, un protocole et une vitesse de transmission élevée (100mb/s).
2- Les Métropolitains MAN (Metropolitan Area Network) : qui correspondent à une interconnexion de quelques bâtiments (Série de réseaux locaux) se trouvent dans une ville (Campus).Ils fonctionnent à des vitesses moyennes et élevées.
3- Les réseaux étendus WAN (Wide Area Network) : destinés à transporter des données à l'échelle d'un pays ou de la planète entière. Ces réseaux peuvent être terrestres (utilisation d'infra - structure au niveau : câble, fibre,...) ou satellite (mise en place d'engins spatiaux pour retransmettre les signaux vers la terre).

IV- Topologie des réseaux

1- Définition de la topologie

C'est une organisation physique et logique d'un réseau (c'est la manière d'interconnecter les nœuds du réseau). L'organisation physique concerne la façon dont les machines sont connectées (Bus, Anneau, Étoile, Maillé, Arborescence, ...). La topologie logique montre comment les informations circulent sur le réseau. On peut différencier les réseaux selon leurs structures ou leurs topologies logiques. On distingue ainsi deux classes de réseaux : **[Ric 94]**

-ceux en **mode de diffusion**
-ceux en **mode point à point**

le jeton émet des données qui font le tour de l'anneau. Lorsque les données reviennent à la station qui les a envoyées, elle les élimine et passe le jeton à son voisin, et ainsi de suite…
Cette topologie est utilisée par les réseaux Token Ring et FDDI.

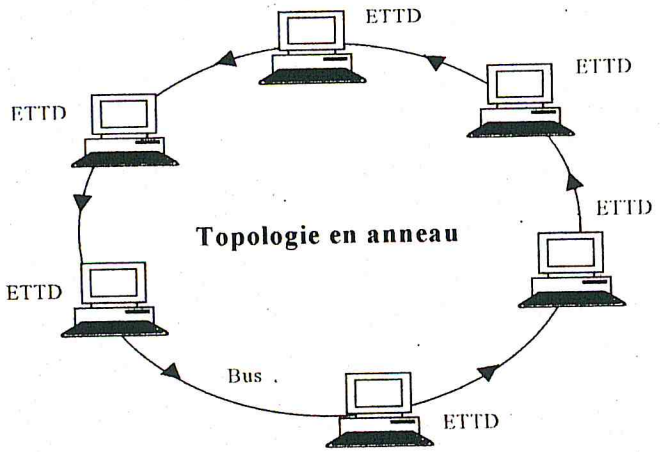


Figure 3 : Topologie en anneau

Les avantagee de la topologie :
-Accès égalitaire à toutes les stations
-Performances régulières même avec un grand nombre de stations
Les inconvénients :
-Une panne d'ordinateur peut affecter l'anneau
-Problème (panne) difficile à isoler
-La reconfiguration du réseau peut interrompre son fonctionnement.

2-3- Topologie en étoile

C'est la topologie la plus courante, dans un réseau en étoile tous les composants (ETTD) sont reliés à un même point central et l'information ne va que de l'émetteur vers le récepteur en transitant par ce point central (**mode point à point**). On trouve typiquement un switch au niveau du nœud central.
Cette topologie est utilisée par les réseaux Ethernet 10 et 100 Base T et par le 100 VG AnyLAN.

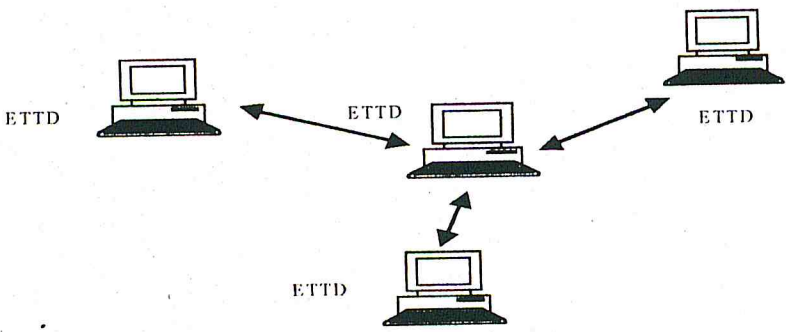


Figure 4 : Topologie en étoile

Les avantages :
-Ajout de station facile.
-Surveillance et gestion centralisée.
L'inconvénient :
-Si le site central tombe en panne tout le réseau est hors-service

2-4- La topologie maillée

Les ETTD sont reliés entre eux de telle sorte que plusieurs chemins soient possibles pour transporter l'information d'un ETTD à un autre. Ce type de réseau peu être plus ou moins

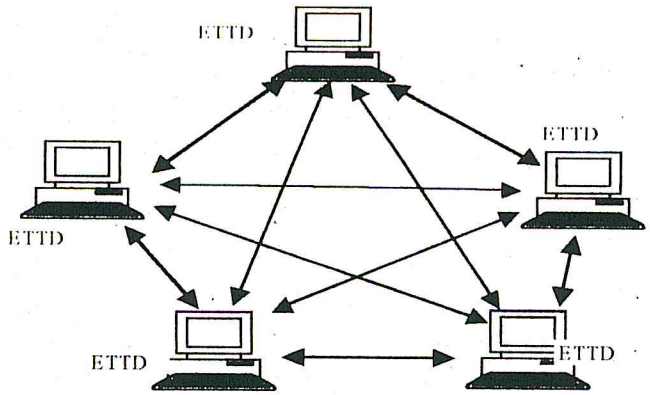


Figure5 :Topologie maillée

fortement maillé. Les ordinateurs du réseau Transpac sont disposés en configuration fortement maillée.

V- Les raccordements physiques

1- Les principaux supports physiques utilisés (câblage)

1-1- Support métallique

Reposent sur la propriété de conductivité électrique des métaux (cuivre, bronze, ...) parmi ces supports :

Paire de fils torsadés :

- constituée d'une paire de fils électriques agencés en spirale
- convient à la transmission analogique et numérique
- adaptée à la transmission d'information de courte distance
- support le plus utilisé : téléphonie
- performances: débits courant: 1 Mbit/s, 4 Mbit/s, 10 Mbit/s et 16 Mbit/s
- adoptée comme support physique des réseaux locaux informatique comme "Token ring"

Câble coaxial : – câble fin (Ethernet fin) – gros câble (gros Ethernet).

- constitué de deux conducteurs cylindriques de même axe, séparés par un isolant
- convient à la transmission numérique et analogique
- adaptée à la transmission d'information de longue distance
- les câbles coaxiaux offerts sur le marché sont :

câble 50 du type Ethernet, le câble de 75 du type CATV

- performances : débits courants : 2 Mbit/s à 100 M bit/s

1-2- Fibre optique

La transmission se fait par propagation d'un rayon lumineux dans une fibre de verre.

- on utilise un faisceau optique modulé
- permet une très large bande passante (de l'ordre de 1GHz pour 1 Km)
- permet une très bonne qualité de la transmission
- performances : Débit courant : 140 Mbit/s

1-3- Transmission terrestre sans support matériel

(a) Courte distance
- rayons infrarouges et rayons laser
- transmission numérique

(b) Longue distance
- faisceaux hertziens, ondes radios magnétiques
- problème des interférences
- partage de la bande passante admissible (2-40 GHz)
- application : réseaux de radio télévision

1-4- Les satellites de communication :

- ondes à très hautes fréquences
- répéteurs à transposition de fréquences
- satellites géostationnaires (36000 Km par rapport à l'équateur)
- bandes de fréquences utilisables (4/6 GHz, 12/14 GHz)
- faible taux d'erreurs
- problèmes de confidentialité (cryptographie)

1-5- Hertzienne

Utilise des ondes radio – électriques. La propagation se fait par ligne droite (radio, télé, ...). Pour permettre des liaisons grandes distance, on utilise des satellites.
Avantage : liaison grande distance. Pas de câblage.
Inconvénien : affaiblissement des signaux et le temps de propagation est de 260 ms pour un aller – retour. Le niveau du signal électronique ou lumineux, représenté par un 1 ou un 0.

2- Les équipements téléinformatiques

2-1- Le HUB (Host Unit Broadcast)

Appeler aussi **concentrateur** est un dispositif de niveau 1. Le HUB assure :
-la communication entre les stations comme si elles étaient reliées à un bus mais physiquement la topologie est de type étoile.
- se contente de faire circuler les signaux dans les deux sens.
- la régénération et la répétition du signal.
- la détection et la notification des collisions.

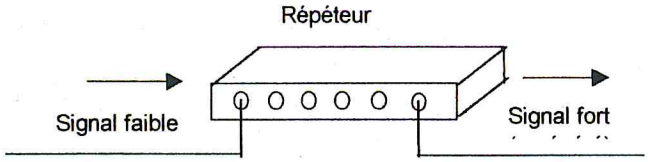
Pour les réseaux sur paires torsadées, chaque station est reliée à un HUB (pivot) par deux paires torsadées (une pour l'émission et une pour la réception).
Il est possible de monter des HUB en cascade, mais il ne doit pas y en avoir plus de quatre.

Lorsqu'une station transmet, le HUB répète le signal et le diffuse à toutes les stations. Quand plusieurs stations émettent en même temps, le HUB détecte de l'activité sur plus d'une entrée, et détecte donc une collision. Il génère alors un signal de présence de collision (CP) qu'il diffuse à toutes les stations tant qu'il détecte de l'activité sur une ligne d'entrée.

Remarque :
Le câble coaxial Ethernet est le seul standard LAN qui n'utilise pas du HUB.

2-2- Le répéteur (repeater)

Un répéteur est un dispositif de réseau qui opère au niveau de la couche Physique, car il ne nécessite aucune information d'adressage de la trame de données. Sa tâche consiste à répéter et renforcer un signal d'un port vers les autres ports auxquels il est connecté. Il n'assure ni tâche de filtrage, ni d'interprétation, et ne fait que répéter (régénérer) les signaux, les transmettant dans toutes les directions.



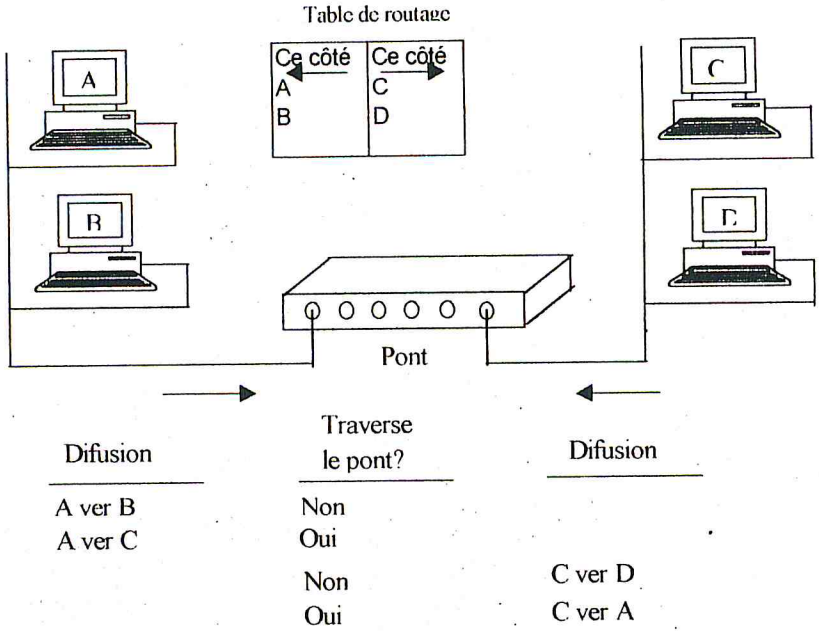
Un répéteur régénére un signal faible,

Le principal objectif d'un répéteur est de permettre au réseau de s'étendre au-delà des limitations de distances dues au média de transmission.
Les fonctions principales du répéteur sont, de synchroniser, d'amplifier, et de remettre en forme les signaux. Le répéteur est indépendant du protocole utilisé, il peut effectuer la transition vers un autre support physique.
Avantage : peu coûteux et simple.
Inconvénient: n'interconnectant pas des réseaux dissemblables (TokenRing et Ethernet)
Certains peuvent servir dans l'interconnexion de segments utilisant des trames de données similaires, mais un câblage différent.

2-3- Le pont (bridge)

Un pont est un dispositif de connectivité qui fonctionne au niveau de la couche Liaison de données. Le pont envoie une trame en direction de ses destinataires sans la transmettre simultanément aux segments auxquels elle n'est pas destinée. Le pont peut donc filtrer le trafic du réseau en le réduisant. Le pont filtre les trames reçues en suivant et contrôlant les adresses du niveau sous-couche MAC (Media Acces Control) de la couche Liaison de données. Un pont peut remplir les mêmes fonctions qu'un répéteur, y compris l'extension des distances de câblage et la liaison de types de câble dissemblables. De plus, un pont peut améliorer les performances et réduire le trafic en divisant le réseau en segments plus petits.
En local, un seul pont est nécessaire.
A distance, les informations passent par les lignes téléphoniques. Un pont est nécessaire à chaque extrémité de la ligne (un sur chaque LAN). On peut connecter plusieurs ponts en cascade jusqu'à un maximum de quinze, sur une épine dorsale en gros câble coaxial.
Avantages :
- simple et peu coûteux,
- lit la source et la destination de chaque paquet,
- peut connecter des réseaux de vitesses différentes,
- peut collecter des statistiques.

Inconvénients :
- pas adapté aux WAN comportant diverses topologies
- ne fournit pas d'isolation d'erreurs.



une conféguration de pont simple

2-4- Le routeur (router)

Un routeur est un dispositif de niveau 3, c'est à dire qu'il intervient aux niveaux 1 à 3. Il est un pont intelligent pour les gros réseaux (il peut réaliser des connexions sur réseau téléphonique au moyen de modems). Les ponts connaissent les adresses de tous les ordinateurs du réseau situés de chaque côté et peuvent faire suivre les messages en conséquence. Mais les routeurs en savent encore davantage sur le réseau. Un routeur connaît non seulement les adresses de tous les ordinateurs du réseau, mais également les autres ponts et routeurs du réseau, et peut choisir le meilleur chemin pour envoyer un message. Un routeur peut écouter le trafic de tout l'ensemble du réseau pour voir quelles en sont les parties les plus actives et donc peut décider de faire suivre un message par une route moins encombrée. Les routeurs sont plus lents et plus chers que les ponts, mais, pour les grands réseaux sont rentables.
Le routeur est un dispositif de base d'Internet. Il lit l'adresse IP des paquets qu'il reçoit, et re-expédie ces derniers au mieux, en s'inspirant des données contenues dans sa **table de routage**. Cette dernière est mise à jour régulièrement, en fonction des données que le routeur reçoit des routeurs voisins en utilisant des protocoles routable (TCP/IP, IPX/SPX).

2-5- La passerelle (gateway)

Une passerelle est un dispositif de niveau 4, c'est à dire qu'elle intervient aux niveaux 1 à 4. Il s 'agit d'un routeur super-intelligent. Les passerelles sont conçues pour relier des types de réseaux radicalement différents (topologie, protocole, système d'exploitation...) ; Ce qu'elles font en convertissant les messages d'un format de réseau en un autre dans les deux sens.
Interconnexion des réseaux hétérogènes tels que X25 à TCP/IP.
Une passerelle effectue des modifications au niveau de la couche transport et éventuellement au-dessus.