REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE A/MIRA DE BEJAIA
FACULTE DES SCIENCES EXACTES
DEPARTEMENT D'INFORMATIQUE

Mémoire de Fin de Cycle

En vue de l'obtention du diplôme de Master Professionnel en Informatique
Option : Administration et sécurité des réseaux

Thème
Développement d'une solution de gestion réseau basée sur le protocole SNMPv3

Présenté par :

GADOUCHE Hania

HAMMOUCHE Feriel

Devant un jury composé de :

Président : Mr OMAR Mawloud : Maître de conférences B

Examinatrice : Mlle KHOULALEN Nadjet : Maître-assistant A

Examinateur : Mr TOULOUM Karim : Maître-assistant A

Encadreur : Mr FARAH Zoubeyr : Maître-assistant A

Année universitaire : 2011/2012

Remerciements

Nos vifs remerciements vont d'emblée à Dieu le tout puissant pour son aide gracieuse.

Nous remercions également nos chers parents pour tous les sacrifices consentis à notre égard et leur énorme soutien.

Tous nos amis (es)

Notre encadreur, ***Mr FARAH Zoubeyr*** *qui nous a encadrées et orientées tout au long de ce projet.*

Les membres de la commission qui jugeront notre travail.

Tous nos enseignants et les membres du département informatique de l'université ***ABDERRAHMANE MIRA.***

Dédicaces

Ce modeste travail est dédié à

A mes chers parents

A mon frère et à ma sœur

A mes amis(es),

Particulièrement Meriem & Sonia.

Feriel

Ce modeste travail est dédié à ma famille ainsi qu'aux personnes qui sont chères à mon cœur.

Hania

Table des matières

Liste des figures

Liste des figures

Liste des tableaux

Glossaire

Glossaire

AES : Advanced Encryption Standard.

API : Application Programming Interface.

ASN.1 : Abstract Syntax Notation 1.

ATM : Asynchronous Transfer Mode.

CMIP : Common Management Information Protocol.

CMIS : Common Management Information Service.

CMISE : Common Management Information Service Element.

DES : Data Encryption Standard.

GUI : graphical user interface.

IAB : Internet Activities Board.

IETF : The Internet Engineering Task Force.

IP : Internet Protocol.

MD5 : Message Digest 5

MIB : Management Information Base.

MSA : Managed System and Agents.

NMS : Network Managment Systems.

OID : Object Identifier.

OMG : Object Management Group.

OMT : Object Modeling Technique.

OOSE : Object Oriented Software Engineering.

OSI : Open Systems Interconnection.

SHA-1 : Secure Hash Algorithm.

SMFA : Specific Management Function Area.

SMI : Structure of Management Information.

SNMP : Simple Network Management Protocol.

SSH : Secure Shell.

SSL : Secure Sockets Layer.

TCP/IP : Transmission Control Protocol / Internet Protocol.

UDP : User Datagram Protocol.

UML : Unified Modeling Language.

USM : User-based Security Model.

VACM : View- based Access Control Model.

Introduction générale

Les réseaux informatiques, composés d'équipements interconnectés, ont aujourd'hui beaucoup plus d'importance qu'ils avaient il ya quelques années. En effet, les entreprises dès leur création n'hésitent pas à mettre en place un réseau informatique pour faciliter la gestion de leur l'infrastructure.

Les technologies utilisées pour transmettre des données d'un ordinateur à un autre font appel à un grand nombre de composants. L'objectif premier de la transmission de données est de permettre aux différents matériels et aux différents systèmes d'exploitation de communiquer et de se comprendre. Certaines technologies, qui peuvent paraître dépassées, sont encore très présentes dans les entreprises. De plus, elles constituent bien souvent le fondement des techniques actuelles.

L'administration à distance a fait ses preuves en ce qui est des tâches de l'administrateur, ceci pourrait lui éviter le déplacement sur chaque équipement constituant le réseau, en particulier dans le cas d'une grande entreprise qui pourrait en compter par centaines.

L'administrateur doit veiller au bon fonctionnement du réseau administré, ce qui signifie que celui-ci doit être stable et dans une parfaite harmonie.

Tout l'art de l'administrateur est de faire en sorte que le réseau puisse fonctionner de manière autonome, de façon à minimiser les interventions manuelles.

Plusieurs solutions d'administration réseau ont été proposées, on cite CMIS/CMIP concernant le modèle OSI, Telnet, SSH, et Interface Web ainsi que SNMP pour TCP/IP.

Contrairement aux autres protocoles cités précédemment, SNMP propose une interface de transaction commune à tous les matériels, donc il représente une solution assez simple et appropriée pour une bonne administration réseau.

L'application développée durant ce travail permet la gestion d'équipements réseaux, tels que les routeurs, les switchs et les ordinateurs. Et ce en consultant leurs états ou en effectuant des opérations de mises à jour sur ceux-ci.

Ce mémoire comprend quatre chapitres.

Le premier chapitre est consacré à l'introduction de l'administration de réseaux, définissant les tâches principales de l'administrateur ainsi que les protocoles de gestion existants dans ce domaine.

Le deuxième chapitre définit le protocole SNMP, ses versions, son architecture ainsi que ses fonctionnalités.

Le troisième chapitre consiste à définir les besoins qui nous ont poussées à élaborer cette application et par conséquent la modélisation en UML de cette dernière.

Enfin, nous conclurons ce mémoire par un chapitre de présentation des différentes fonctionnalités de l'application ainsi que les outils qui ont permis à la réalisation de celle-ci.

Chapitre 1

Introduction à l’administration réseau

1. Introduction

Actuellement, l'évolution des réseaux informatiques pose un problème majeur. En effet, le nombre de nœuds ne cessant de s'étendre, il est important de pouvoir gérer tous ceux-ci le plus facilement possible, mais en conservant certaines contraintes techniques. Afin d'exploiter au mieux les ressources disponibles Il faut souvent avoir recours à des techniques d'administration.

Le terme administration réseau recouvre l'ensemble de fonctions et de tâches réalisées par l'administrateur réseau, nécessaires à l'exploitation, la sécurité, le suivi et l'entretien du réseau. II est important de pouvoir initialiser de nouveaux services, installer de nouvelles stations raccordées au réseau, superviser l'état du réseau global et de chacun de ses sous ensembles.

2. Définition de l'administration réseau

L'administration est la fonction principale d'un administrateur réseau. Elle consiste à mettre en place, maintenir et organiser l'infrastructure du réseau, mais aussi :

- Installer et maintenir les services nécessaires au fonctionnement du réseau.
- Assurer la sécurité des données internes au réseau (particulièrement face aux attaques extérieures).
- S'assurer que les utilisateurs "n'outrepassent" pas leurs droits.
- Gérer les logins (noms d'utilisateurs, mots de passe, droits d'accès, permissions particulières, . . .).
- Gérer les systèmes de fichiers partagés et les maintenir.

3. Les ressources gérables [OLW]

La gestion réseau est la tâche quotidienne de tout administrateur réseau : il s'agit de vérifier le fonctionnement optimal de chaque matériel, de paramétrer celui-ci et de le mettre à jour régulièrement. Ainsi, dans les réseaux informatiques actuels, la gestion (ou *management*) est un problème quotidien.

On peut distinguer plusieurs niveaux de gestion, et ce à partir des différents éléments constituant le réseau :

- **La gestion de l'infrastructure réseau :** concerne la gestion de tout ce qui est équipement relié au réseau, le logiciel y résidant ainsi que les logiciels embarqués qui constituent les différents réseaux de l'entreprise.

  Un équipement peut être un routeur, un concentrateur, un répéteur, une passerelle, ou encore un modem.
- **La gestion d'ordinateurs :** concerne tout ce qui se réfère à la gestion de points d'accès au réseau ; la gestion des stations terminales ainsi que les logiciels qui y sont supportés, à savoir, un système d'exploitation réseau mais aussi les applications et services de communication mis à la disposition des usagers.

4. Domaines fonctionnels de l'administration réseau [NAG]

L'administration réseau est une tâche assez complexe, c'est pour cela que ISO a défini ses activités en cinq axes :

4.1. La gestion d'anomalies ou défauts (Fault Management)

L'objectif de l'administration réseau est d'avoir un réseau opérationnel sans rupture de service (taux de disponibilité à 99,999 % par exemple soit quelques secondes d'indisponibilité par an), ce qui définit une certaine Qualité de Service (QoS) qui se distingue sur plusieurs critères, du point de vue de l'utilisateur final, notamment la disponibilité, la performance (temps de réponse), la fiabilité et la sécurité offerte par l'opérateur. L'administrateur doit être en mesure de localiser le plus rapidement possible tout dysfonctionnement, toute panne ou défaillance interne ou externe du système.

Ces pannes pouvant être d'origine interne résultant d'un élément du réseau ou d'origine externe résultant de l'environnement du système (coupure d'un lien publique). La panne est localisée par un diagnostic des alarmes émises par le réseau, afin que l'administrateur puisse journaliser les problèmes et tenter de les résoudre.

Les phases de traitement d'un défaut [BCL]

1 - Détection d'un fonctionnement anormal : messages d'erreur (quoi, qui (où), quand), et tests (de contrôle routinier, de diagnostic).

2 - Localisation/diagnostic : exploitation de l'historique (suite d'évènements, ensemble d'évènements), tests de diagnostic.

3 - Réparation : reconfiguration, remplacement.

4- Vérification : retour au fonctionnement normal.

4.2. La gestion des performances (Performance Management)

Consiste à analyser de manière continue les performances du réseau afin de le maintenir dans un état acceptable. La gestion se fait en trois étapes : Tout d'abord, des variables contenant des informations de performances du réseau sont récupérées. Parmi celles-ci on peut citer le temps de réponse d'une station utilisateur ou encore le taux d'occupation d'un segment réseau. Une fois ces variables obtenues, elles sont analysées ; si elles dépassent un seuil de performance fixé préalablement, une alarme est tout de suite envoyée à l'administrateur du réseau, pour régler le problème au plus vite. Ces variables de performances sont réactualisées à court intervalle de temps dans le but d'une réaction rapide au moindre embryon de baisse de performance. La gestion des performances permet donc une évaluation du comportement des ressources et un contrôle de l'efficacité des activités de communication.

4.3. La gestion de la configuration réseau (Configuration Management)

Effectue un suivi des différentes configurations des éléments présents sur le réseau ; matérielles et logicielles pour en optimiser l'utilisation. Elle stocke dans une base de données les versions de systèmes d'exploitation des logiciels installés sur chaque machine du parc réseau. Il est important que chaque équipement, chaque compteur… soit parfaitement identifié de façon unique à l'aide d'un nom ou identificateur d'objet OID (Object Identifier).

4.3.1. Rôle principal

- Inventaire de ressources,
- Initialisation d'équipements,
- Gestion de noms et d'adresses,
- Mise à jour des paramètres de ressources.

4.3.2. Procédures

- Collecter les informations,
- Contrôler l'état du système,
- Sauvegarder l'historique ("log"),
- Présenter l'état du système synoptique.

4.4. La gestion de la sécurité (Security Management)

Concerne les contrôles d'accès au réseau, la confidentialité des données qui y transitent, leur intégrité ainsi que leur authentification.

Cette gestion est nécessaire pour protéger le réseau contre : un dysfonctionnement, une inadvertance, une malveillance, une attaque passive telle que l'écoute de messages ou l'observation du trafic, ou une attaque active comme la duplication/modification de messages ou de services.

Afin d'assurer la sécurité, l'administrateur peut avoir recours à :

- **Une base de données spécifique**

- Security MIB : rassemble (protège) les informations utilisées pour assurer la sécurité.

- **Enregistrement de l'activité des utilisateurs**

- Les évènements significatifs, les actions interdites ou sensibles.

- **Filtrage ("firewall")**

- Tri des flux de données circulant entre deux parties du réseau.
- Filtrage sur les adresses IP, le sens d'entrée/sortie, le type du protocole et le numéro de port.
- Les applications accessibles doivent être protégées.

- **Authentification de :**

- L'utilisateur de service (avec un mot de passe) et de l'émetteur de message (avec la signature).
- La tierce partie.

- **Intégrité du message**

- Fonctions de hachage.

- **Cryptage de messages**

- Système à clé secrète (symétrique) ou publique (asymétrique).
- Cryptage à la source ou par un serveur.

4.5. La gestion de la comptabilité (Accounting Management)

La gestion de la comptabilité a pour but de mesurer l'utilisation des ressources afin de régulariser les accès et instaurer une certaine équité entre les utilisateurs du réseau. Ainsi, des quotas d'utilisation peuvent être fixés temporairement ou non sur chacune des ressources. De plus, la gestion de la comptabilité autorise la mise en place de systèmes de facturation en fonction de l'utilisation pour chaque utilisateur.

La gestion de la comptabilité permet donc l'établissement des coûts d'utilisation ainsi qu'une facturation de l'utilisation des ressources établie en fonction du volume et de la durée des transmissions.

Exemple

- Nombre d'octets transmis, durée de connexion.

5. Types de décisions administratives [WLY]

On distingue trois différents types de décisions administratives :

5.1. Décisions opérationnelles : décisions à court terme, concernant l'administration au jour le jour et opérations temps réel sur le système, servent à maintenir le réseau en état opérationnel.

5.2. Décisions tactiques : décisions à moyen terme, concernant l'évolution du réseau et l'application des politiques de long terme, permet l'optimisation du fonctionnement du réseau, et ce à partir des résultats des fonctions de statistiques et de tests dans des périodes significatives. Des changements seront à faire si nécessaire après analyse des résultats obtenus.

5.3. Décisions stratégiques : décisions à long terme, concernant les stratégies pour le futur du réseau en exprimant les nouveaux besoins et désirs des utilisateurs.

6. Principe général d'administration [OLW]

Un réseau informatique se compose d'un ensemble d'objets qu'un système d'administration doit surveiller et contrôler.

Les systèmes de gestion de réseau (Network Managment Systems (NMS)) possèdent deux éléments de base:

- Des "Manager" qui permettent aux gestionnaires du réseau de le contrôler et d'y réaliser des actions.
- Des "Agents" qui sont l'interface entre les objets des périphériques gérés et le Manager.

Les outils d'administration de réseau sont le plus souvent basés sur une architecture de type client/serveur. Dans celle-ci, le client, appelé « manager », est le logiciel qui centralise les informations en provenance de l'ensemble des équipements de réseau par l'intermédiaire d'« agents » qui peuvent agir pour son compte.

Le « manager » interroge, envoie des commandes de contrôle ou reçoit des notifications de la part des agents qui représentent les composants à administrer. Pour cela, il faut que les deux parties utilisent :

- **Un dialogue standardisé** : c'est le rôle du protocole d'administration de réseau qui formalise les échanges possibles. Ce protocole peut être standard (CMIP dans le modèle OSI et SNMP dans TCP/IP).

  Le manager peut émettre des commandes de type GET qui permettent de récupérer l'état des objets représentés par l'agent, tandis que les commandes de type SET permettent de le changer. Un agent peut aussi émettre des notifications, pour avertir le manager des changements d'états par exemple.

- **Une représentation commune des composants administrés** : réalisée au moyen d'une base d'information. Celle-ci constitue la description formelle et commune des objets administrés.

  La figure suivante représente la structure fonctionnelle d'une administration réseau :

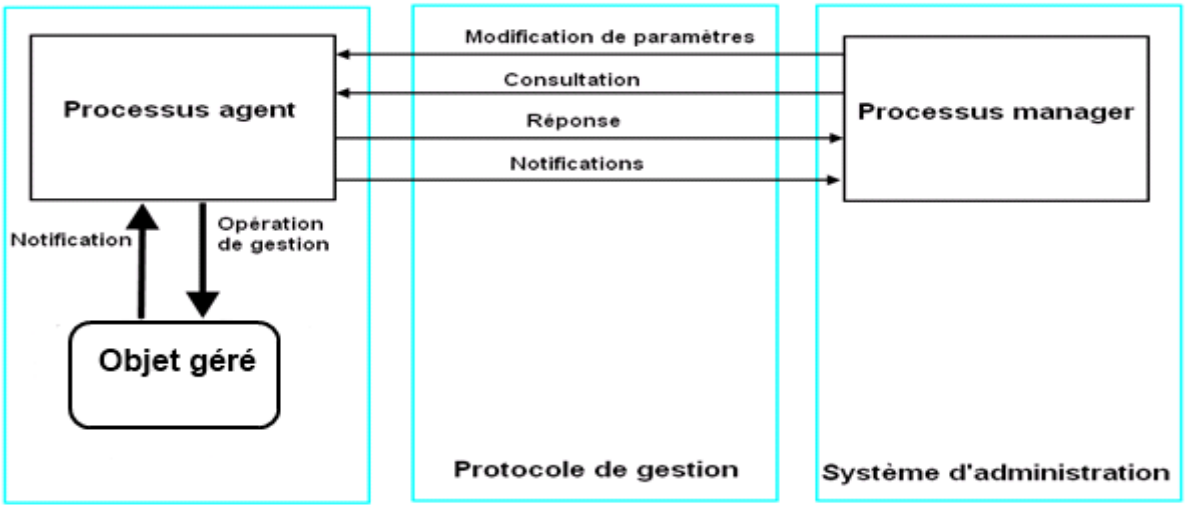

Figure 1.1 : Structure fonctionnelle d'une administration réseau

7. Protocoles de gestion existants [NES]

Il existe plusieurs protocoles de gestion réseau ; des protocoles basés sur le modèle OSI et d'autres basés sur TCP/IP, norme protocolaire utilisée dans les réseaux de type Internet.

7.1. Protocole pour la gestion des réseaux dans le modèle OSI : CMIS/CMIP

La fonction fondamentale du CMISE (Common Management Information Service Element) est l'échange d'informations de gestion entre deux administrateurs ou entre administrateur et entités agents. Il est spécifié en deux parties : CMIP et CMIS.

- **CMIS :** (Common Management Information Service) définit un système qui fournit des services d'information de gestion de réseau.

- **CMIP :** Protocole de l'information de gestion commune (Common Management Information Protocol) est un protocole d'OSI utilisé avec les services d'information de gestion commune(CMIS), il est basé sur le principe de notification et d'événements, il est sécurisé et peut fonctionner sur des réseaux hétérogènes.

CMIP n'indique pas la fonctionnalité de l'application de gestion réseau, il définit seulement le mécanisme d'échange d'information entre objets contrôlés et n'indique pas comment l'information doit être employée ou interprétée.

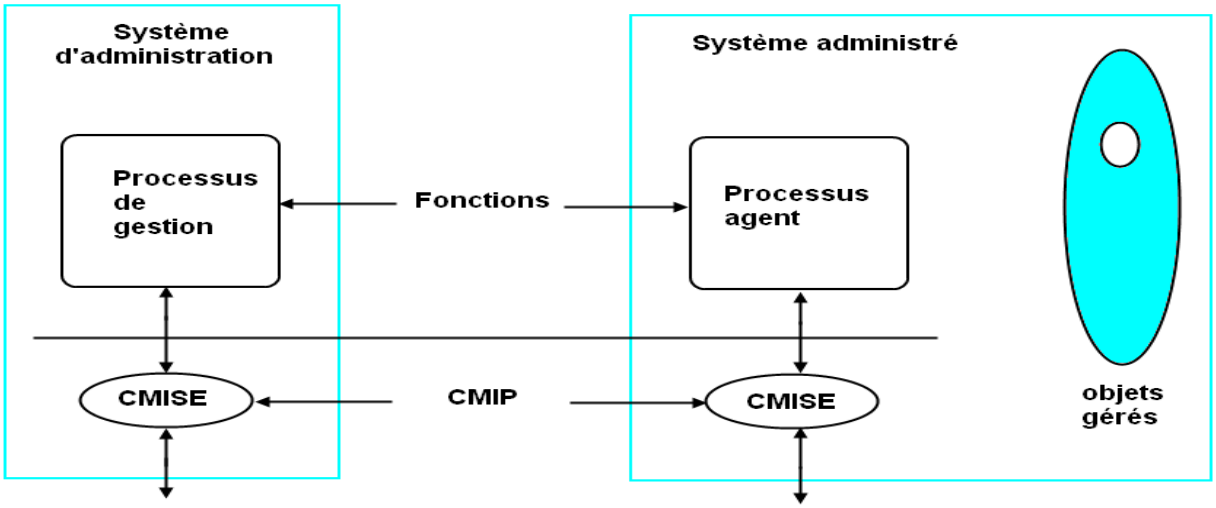

Figure 1.2 : Schéma organisationnel d'un système de gestion basé sur CMIS

Le protocole CMIP se base sur les trois modèles suivants :

1. **Modèle d'architecture MSA (Managed System and Agents)** : définit l'architecture de la gestion du protocole CMIP et la notion de systèmes gérés et gérants.
2. **Modèle d'information** : définit le modèle de représentation des informations de gestion.
3. **Modèle fonctionnel SMFA (Specific Management Function Area)** : définit des domaines fonctionnels d'administration et leurs relations.

7.2. Protocoles pour la gestion des réseaux dans le modèle TCP/IP

7.2.1. Telnet

Le premier protocole historique est Telnet. Ce protocole est largement utilisé pour le contrôle à distance de matériels réseaux. La conception est très simple : une fois que l'on est connecté à la machine distante, les touches tapées au clavier sont directement transmises à la machine distante et Telnet nous renvoie les réponses de ladite machine.

Généralement, la machine distante commence la transaction par demander un mot de passe d'accès, puis donne accès à un Shell sur lequel les commandes peuvent être lancées.

Telnet n'est pas un protocole fournissant une interface commune à tout le matériel réseau. Chaque constructeur inclut son propre gestionnaire Telnet personnalisé, et la gestion du réseau n'est donc pas uniforme suivant le type de matériel.

Telnet assure intrinsèquement la fiabilité de la transaction du fait qu'il soit basé sur le protocole TCP, toutefois la communication entre l'administrateur et le matériel n'est pas cryptée et la seule sécurité apportée est l'authentification initiale.

7.2.2. SSH (Secure Shell)

Le protocole SSH comble la lacune de sécurité concernant Telnet en cryptant la transaction via le protocole SSL (Secure Sockets Layer). Il permet également d'effectuer des transferts de fichiers entre les deux hôtes. Toutefois, l'interface reste propre à chaque matériel et ne permet pas d'effectuer des transactions parallèles ou une gestion uniforme de ceux-ci.

7.2.3. Le protocole SNMP

SNMP (Simple Network Management Protocol) est un protocole standardisé qui permet à l'administrateur réseau, depuis un poste de contrôle central, d'obtenir et de modifier divers éléments de configuration d'équipements actifs et de logiciels. SNMP permet de vérifier l'état des périphériques, mais il peut aussi permettre de suivre leur utilisation dans le temps et observer l'état général de son réseau. Il utilise le concept d'application Client/Serveur bien connu dans le monde IP. Son plus grand avantage est l'interface de transaction qu'il propose, commune à tous les matériels.

Dans ce qui va suivre, notre étude se portera sur ce protocole.

7.3. Comparaison des protocoles

Chacun des protocoles présentés a ses propres objectifs et en conséquence ses propres avantages :

- Telnet est un standard en matière de communication à distance et son objectif est d'exécuter par l'intermédiaire d'une interface des commandes de gestion.
- SSH est similaire à Telnet, tout en offrant une couche supplémentaire permettant de crypter le contenu de la transaction.
- SNMP est un protocole proposant une interface commune à tous les matériels et donc une homogénéité accrue.
- SNMP apporte une sécurité améliorée par rapport aux autres protocoles.
- CMIP/CMIS de OSI est proposé en tant que protocole de concurrence à SNMP dans la suite de TCP/IP.

8. Conclusion

Les réseaux sont devenus un pilier de l'économie mondiale. Les besoins et les enjeux de ces technologies ne cessant d'augmenter a engendré l'évolution de l'administration, afin d'apporter une garantie de fiabilité, de réactivité et d'adéquation des moyens.

L'évolution de ces réseaux nécessite la mise en place de protocoles et services de gestion dans le but de faciliter la tâche aux administrateurs. Ces protocoles doivent être évolutifs et prendre en compte uniformément les évolutions des différents matériels.

Le chapitre suivant traite de façon approfondie le protocole le plus standardisé pour la gestion de réseaux, à savoir le protocole SNMP.

Chapitre 2

Le protocole SNMP

1. Introduction

La gestion de réseaux informatiques nécessite la mise en œuvre d'une infrastructure simple et efficace, pour ce faire, SNMP a été développé comme standard de gestion de ces derniers. Il permet de contrôler des équipements à distance et ce en accédant entre autre aux paramètres de ceux-ci et de les modifier ou les mettre à jour si nécessaire.

Dans ce chapitre, nous étudions de façon plus approfondie le protocole SNMP, ses différentes versions, son architecture ainsi que son fonctionnement.

2. Présentation [NES]

SNMP est un protocole utilisé dans des programmes de gestion de réseaux informatiques dans le modèle TCP/IP. Son fonctionnement est asymétrique, il est constitué d'un ensemble de requêtes, de réponses et d'un nombre limité d'alertes. La « Station de Gestion » envoie des requêtes à l'agent, qui retourne des réponses. SNMP utilise le protocole UDP [RFC 768].

Le gestionnaire SNMP doit être capable de lire et de modifier les valeurs des variables MIB (Management Information Base) des périphériques qu'il gère.

SNMP est un protocole bâti au dessus d'UDP/IP :

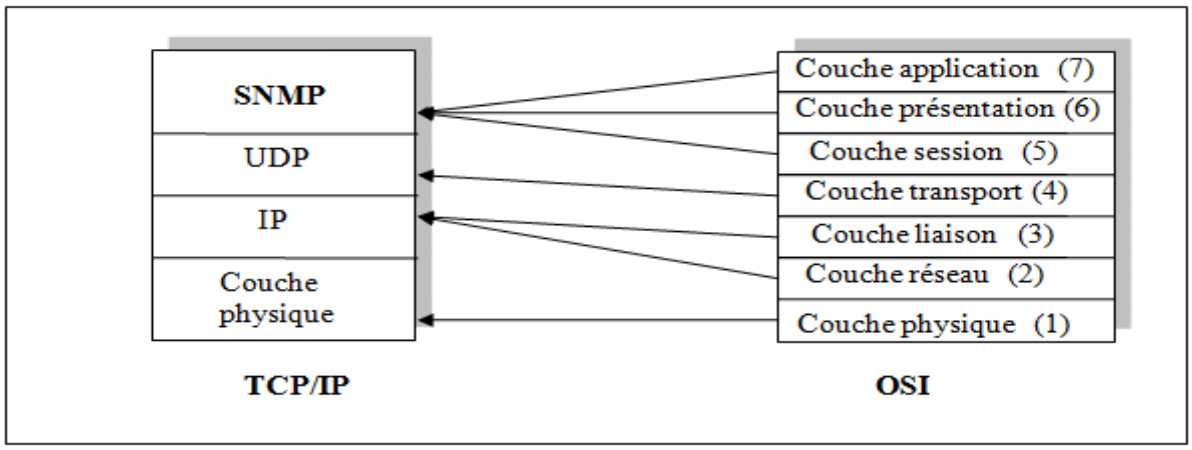

Figure 2.1 : SNMP et la pile de protocoles **[FAT]**

3. Historique [FFO]

La première version de SNMP, SNMPv1, a été conçue à la fin des années 80 et standardisée dans le courant de l'année 1990. Un certain nombre de lacunes persistaient : manque de hiérarchie, faibles performances, sécurité laxiste …etc.

L'ensemble de ces problèmes a entraîné le développement d'une nouvelle version de SNMP, nommée SNMPv2, et dont la conception a commencé en 1993. Toutefois, plusieurs éditeurs ont rejeté les standards proposés, conduisant à la création d'autres normes :

• **SNMPv2p (historique)**: Beaucoup de travaux ont été exécutés pour faire une mise à jour de SNMPv1. Ces travaux ne portaient pas seulement sur la sécurité. Le résultat est une mise à jour des opérations du protocole, de nouvelles opérations ainsi que de nouveaux types de données. Cette version est décrite dans les RFC1441, RFC 1445, RFC 1446, RFC 1448 et RFC 1449.

• **SNMPv2c (expérimental)**: Cette version du protocole est appelée « community string based SNMPv2 ». Il s'agit d'une amélioration des opérations de protocole et des types d'opérations de SNMPv2p qui utilise la sécurité par chaîne de caractères « community » de SNMPv1. Cette version est définie dans les RFC 1901, RFC 1905 et RFC 1906.

• **SNMPv2u (expérimental)**: Cette version du protocole utilise les opérations, les types de données de SNMPv2c et la sécurité basée sur les usagers. Cette version est décrite dans les RFC 1905, RFC 1906, RFC 1909 et RFC 1910.

• **SNMPv2* (expérimental)**: Cette version combine les meilleures parties de SNMPv2p et SNMPv2u, mais les documents qui décrivent cette version n'ont jamais été publiés.

La version la plus utilisée de SNMP est actuellement SNMPv2c, mais la tendance s'inverse avec l'introduction en 1997 de la troisième version du protocole : SNMPv3.

• **SNMPv3:** Cette version apporte essentiellement à la précédente des fonctions de sécurité, ainsi qu'une gestion hiérarchisée, et par conséquent formalise de façon complète le modèle d'administration SNMP.

4. Architecture de SNMP

La figure ci-dessous présente l'architecture générale du protocole SNMP. En soulignant les liens existants entre le gestionnaire, l'agent SNMP et les objets gérés par celui-ci.

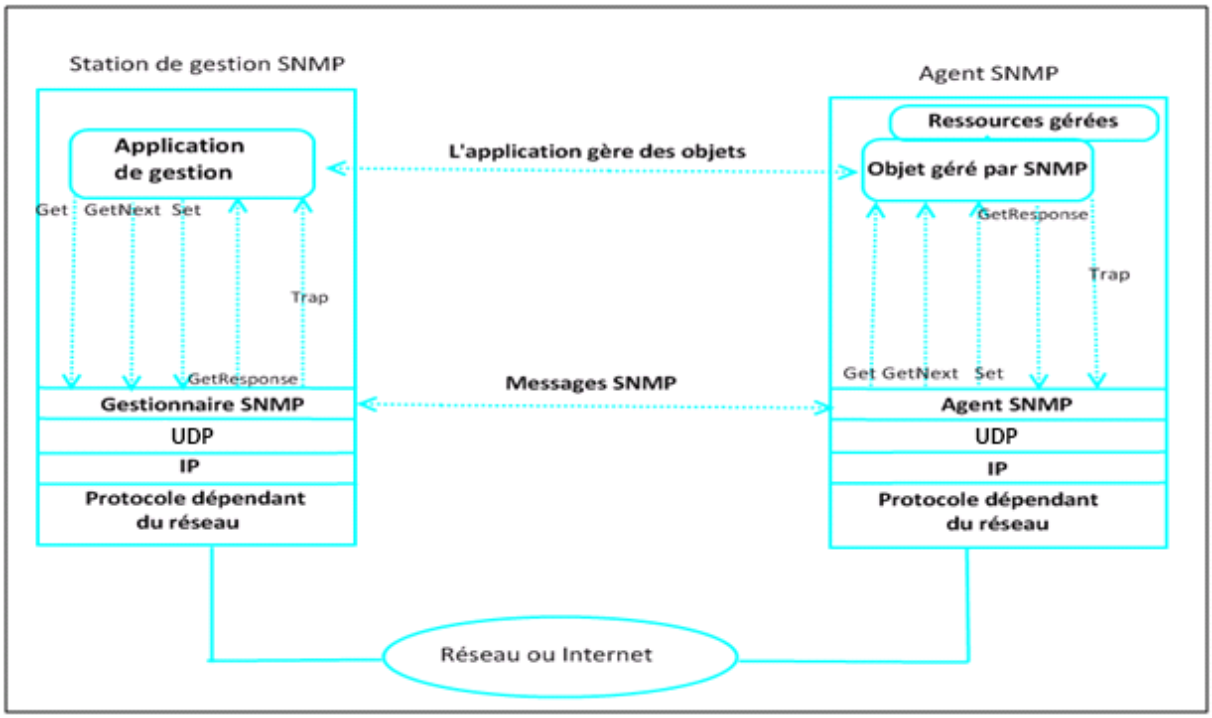

Figure 2.2 : Architecture de SNMP

a. Equipements gérés (Managed devices)

Ce sont des éléments du réseau (ponts, hubs, routeurs ou serveurs …), qui contiennent des "objets de gestion" (managed objects) pouvant être des informations sur le matériel, des éléments de configuration ou des informations statistiques.

b. Station d'administration

Une station d'administration est le périphérique utilisé par l'administrateur pour gérer son réseau, celui-ci doit obligatoirement posséder :

- Des applications spécifiques à l'administration,
- Une interface avec l'administrateur,
- La capacité de pouvoir récupérer des informations des éléments administrés,
- Une base de données obtenue à partir des MIBs des éléments administrés.

La station d'administration NMS (*Network Management Station*) peut envoyer des requêtes à un périphérique afin d'obtenir des informations sur ses paramètres. L'agent du périphérique reçoit la requête et renvoie les informations demandées. Lorsqu'elle reçoit cette réponse, la station NMS peut utiliser les informations de configuration du périphérique afin de déterminer les opérations à entreprendre en fonction de son état.

c. Agent de gestion [NES]

Sur chaque équipement administrable s'exécute un programme serveur : l'agent SNMP. Installé dans des périphériques qui supportent le protocole SNMP (managed devices), il gère les informations relatives à l'équipement qui seront stockées dans une base de données propre : la MIB.

d. MIB (Management Information Base) [FFO]

Il s'agit d'une base de données référençant la liste des objets et des variables associées, des types de données utilisés ainsi qu'un descriptif pour chacune d'entre elles. La base contient éventuellement des types de données personnalisés.

e. Protocole SNMP

C'est un protocole d'échange entre manager et agent.

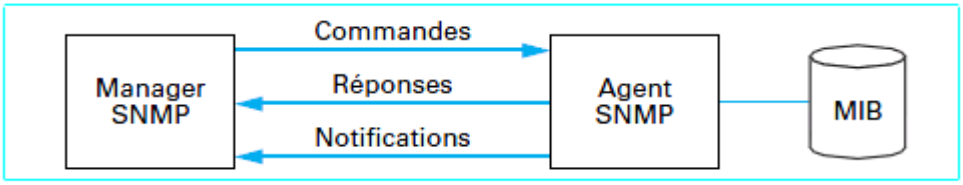

Figure 2.3 : Modèle Agent/Manager

Le manager SNMP lance des commandes de gestion qui peuvent être de consultation ou de mise à jour. L'agent, qui maintient une MIB, applique les commande reçues par le manager et envoie des réponses d'acquittement dans le cas d'une mise à jour ou d'information sur les variables demandées lorsqu'il s'agit d'une consultation. L'agent peut aussi notifier le manager d'évènements ou de pannes sous forme d'alarmes.

- **Utilisation de Proxy [OLW]**

On appelle *proxy* un agent SNMP qui agit pour le compte d'un ou de plusieurs autres équipements non-SNMP mandatés. Il s'agit d'une passerelle de niveau applicatif qui dialogue d'un côté en SNMP et de l'autre avec un autre protocole. Les proxys ont été essentiellement utilisés pour permettre à des équipements préexistants au standard SNMP de s'intégrer dans une solution d'administration SNMP.

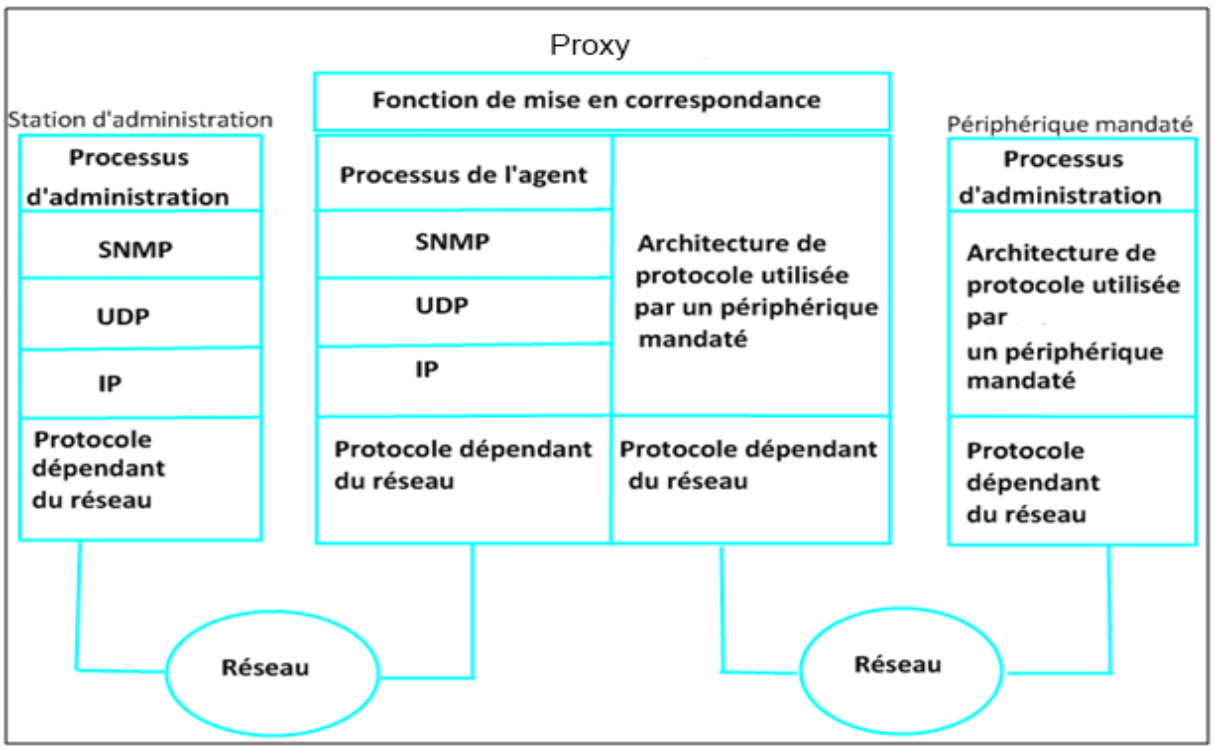

Figure 2.4 : Modèle Agent /manager avec Proxy.

5. MIB [OLW]

Le plus grand avantage du protocole SNMP est probablement celui qui a permis de décrire un grand nombre de composants (réseaux ou autres) de façon standard. Cette description est faite dans la MIB (Management Information Base) : une base de données contenant les informations sur les éléments du réseau à gérer, chaque ressource à gérer correspond à un objet, toutefois il ne s'agit pas d'un objet au sens informatique du terme, mais plutôt un ensemble de variables typées qui peuvent être lues ou mises à jour. La MIB est donc une collection structurée d'objets.

Chaque nœud dans le système doit maintenir une MIB qui reflète l'état des ressources gérées. Une entité d'administration peut accéder aux ressources du nœud en lisant les valeurs de l'objet et en les modifiant.

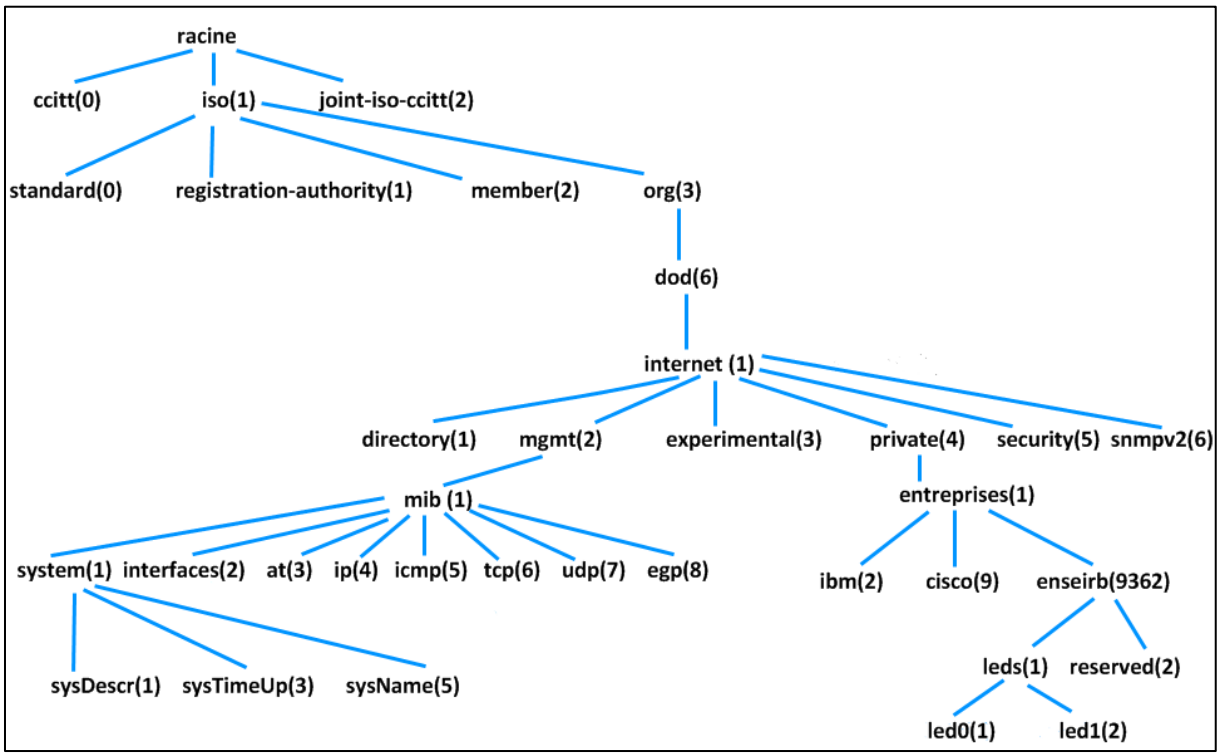

Figure 2.5 : MIB (Management Information Base) **[OLW]**

- La racine de la hiérarchie n'a pas de nom et est désignée simplement par un point.
- **iso (1)** désigne l'Organisation Internationale de Standardisation (*International Organization for Standardization*).
- **org (3)** a été créé par l'ISO à l'intention de divers organismes.

- **dod (6)** a été attribué au ministère de la Défense des États-Unis (Department of Defense).
- **internet (1)** regroupe tout ce qui touche à l'Internet.
- **mgmt (2)** est utilisé pour les standards de l'IAB (*Internet Activities Board*).
- L'arbre de nommage de la figure est infini. Dans l'espace de nommage, seule la partie située sous 1.3.6 est utilisée par SNMP.

La partie supérieure est gérée par l'ISO, qui a prévu un espace de nommage pour le Département de la Défense américain (DoD), organisme à l'origine d'Internet.

a. Nommage des objets [OLW]

Chaque objet de la MIB est identifié par un nom et un OID (*Object Identifier*) qui le décrit de manière unique dans un espace de nommage global et hiérarchique. Ce dernier est représenté par un arbre inversé, dans lequel chaque bifurcation représente un objet. Chaque objet possède un numéro unique qui le situe par rapport au nœud père, ainsi qu'un nom symbolique. Le chemin suivi pour aller de la racine à l'objet constitue l'OID de celui-ci.

Exemple :

Sur la figure ci-dessus, l'objet décrivant le nom d'une machine s'appelle sysName et a un OID défini de la manière suivante :

sysName OBJECT IDENTIFIER ::= { iso(1) org(3) dod(6) internet(1) mgmt(2) mib-2(1) system(1) sysName(5)}

S'il est défini à partir de son père : sysName OBJECT IDENTIFIER ::={system 5}

En abrégé, il est noté .1.3.6.1.2.1.1.5

b. Modules de MIB

Les objets sont décrits par ensembles qu'on appelle « modules de MIB ».

Par abus de langage, on les appelle les MIB. Ils sont classés dans l'espace de nommage sous la branche « internet » :

- **« mgmt (2) »** : contient tous les modules de MIB standard établis par l'IETF.

- **« expérimental (3) » :** contient les modules de MIB établis de manière expérimentale par l'IETF (sachant qu'ils prennent leur forme modifiée et définitive dans des sous arborescences de « mgmt »).

- **« private (4) » :** contient les modules de MIB développés en dehors des standards IETF, par exemple par des entreprises ou des centres de recherche. Les modules de MIB développés sous cette arborescence sont communément appelés « MIB privées ».

○ Plusieurs MIB existent : La MIB2 (RFC1213), et d'autres MIB standard pour des besoins plus spécifiques: ATM MIB (RFC2515), BGB4 MIB (RFC1657), Mail Monitoring MIB (RFC2249), DNS Server MIB (1611)...

○ Les constructeurs ont la liberté de créer leurs propres MIB selon leurs besoins spécifiques (températures, ACL...). Elles apparaissent dans l'arborescence **"private"**).

Exemple de variables de MIB :

Variables MIB	Catégorie	Signification
sysUpTime	système	Durée écoulée depuis le dernier démarrage
ifNumber	interfaces	Nombre d'interfaces réseau
ifMtu	interfaces	MTU d'une interface particulière
ipDefaultTTL	IP	Valeur utilisée dans le champ TTL
ipInReceives	IP	Nombre de datagrammes reçus.
ipForwDatagrams	IP	Nombre de datagrammes acheminés
ipOutNoRoutes	IP	Nombre d'erreurs de routage
ipReasmOKs	IP	Nombre de datagrammes réassemblés
ipFragOKs	IP	Nombre de datagrammes fragmentés
ipRoutingTable	IP	Table de routage IP
icmpInEchos	ICMP	Nombre de demandes d'écho ICMP reçues
tcpMaxConn	TCP	Nombre max de connexions TCP autorisées

tcpInSegs	TCP	Nombre de segments reçus par TCP
udpInDatagrams	UDP	Nombre de datagrammes UDP reçus

Tableau 2.1 : Exemple de variables de MIB

6. Description des objets (SMI) [NES]

La structure SMI (Structure of Management Information) définit les règles de description de l'information et permet d'identifier de façon unique un objet de la MIB gérée par un agent SNMP. Chaque objet possède donc un identificateur unique ou OID (Object ID). SMI s'intéresse aussi à la représentation des données (et leur type) pour chaque objet de la MIB. Un objet de la MIB est déclaré et défini en langage ASN.1 (*Abstract Syntax Notation 1* : langage de représentation de donnée).

Deux versions de SMI coexistent : la première est décrite dans la RFC 1157. La seconde (RFC 2579) est compatible ascendante avec la première et doit être utilisée pour écrire les RFC décrivant de nouveaux modules de MIB.

6.1. Le langage ASN [PRE]

Une application qui échange des données avec une autre a besoin de se faire comprendre par l'application distante. Or, il est courant que deux applications représentent de manière différente les données qu'elles manipulent.
Chaque application sur un site donné dispose donc d'une syntaxe réelle pour représenter l'information.

Une solution à ce problème consiste, pour une application, à transformer sa syntaxe réelle propre en la syntaxe réelle de son correspondant. L'inconvénient d'une telle solution est qu'elle exige de chaque application la connaissance et l'interprétation de toutes les syntaxes pouvant exister.

Une autre solution consiste à utiliser une syntaxe commune à toutes les applications. La syntaxe de transfert définie par ASN.1 est une syntaxe commune et par conséquent va faciliter la communication entre deux applications.

Le principe de fonctionnement permettant à deux applications de communiquer est le suivant :

- Au niveau applicatif, les données devant être échangées entre deux entités d'application sont décrites par une syntaxe indépendante des langages de programmation utilisés, des systèmes d'exploitation et des processeurs ; cette syntaxe porte le nom de syntaxe abstraite.
- A partir de la notation de cette syntaxe abstraite, un outil (compilateur) génèrera d'une part la syntaxe réelle de ces données, et d'autre part un ensemble de fonctions de codage et de décodage (les règles de codage).
- Lors d'une connexion, la couche présentation transformera la syntaxe réelle en un flot d'octets, appelé syntaxe de transfert, à l'aide des règles de codage. Ce flot d'octets est ensuite transféré via une connexion session.

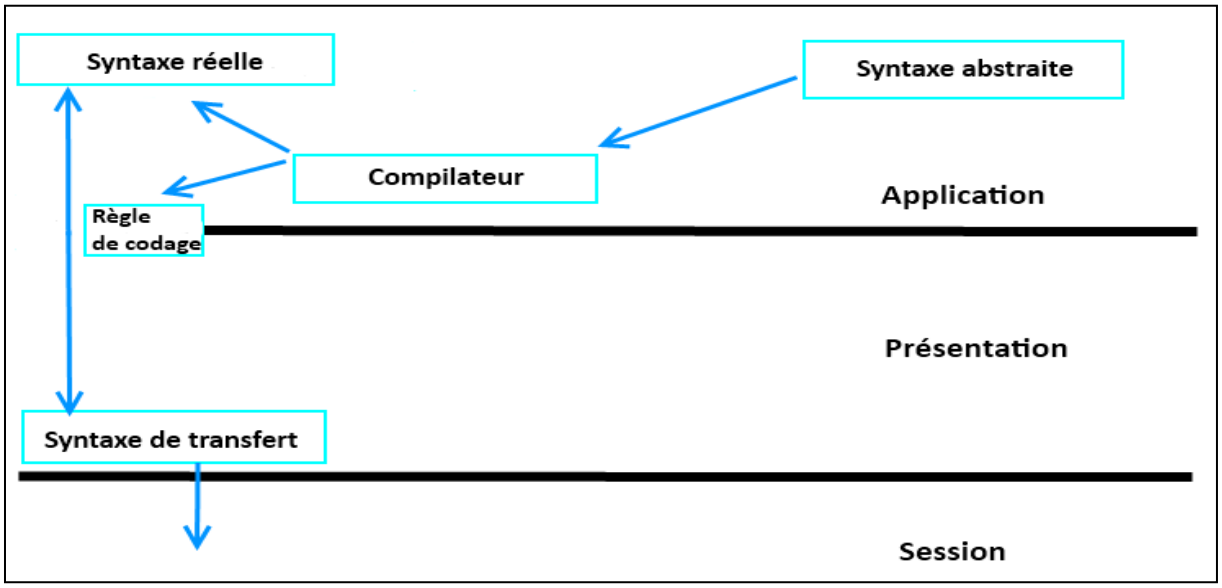

Figure 2.6 : Principe de fonctionnement d'ASN.1 [PRE]

6.1.1. Définition

ASN.1 est un langage utilisé pour décrire de manière abstraite les données échangées par des applications de communication. Il est indépendant des applications, des systèmes d'exploitation et des processeurs.

ASN.1 est un ensemble de types de données de base pouvant être réutilisés pour en construire d'autres. C'est également un ensemble de règles et opérateurs de construction permettant de définir de nouveaux types de données.

- **Les modules**

Un module ASN.1 décrit une collection d'objets.

- **Les objets**

Les objets définis avec ASN.1 peuvent être :

- Des types simples comme INTEGER ou BOOLEAN, des types construits permettant de définir des listes (SEQUENCE) et des tableaux (SEQUENCE OF).
- Des valeurs, c'est-à-dire des objets ayant un type précédemment défini.
- Des macros, qui permettent d'étendre les définitions et définir de nouveaux types.

Par convention, les types commencent par une majuscule, les valeurs par une minuscule et les macros sont entièrement en majuscules. Les commentaires sont précédés de deux tirets. **[OLW]**

6.1.2. Les types

SNMP n'utilise qu'une petite partie du langage ASN.1. Au niveau des types, seuls quelques uns sont utilisés.

- **Les types simples :** le tableau suivant illustre les types simples utilisés par SNMP.

TYPE	**DESCRIPTION**
INTEGER	Entier de 32 bits
OCTET STRING	Chaîne d'octets
NULL	Vide
OBJECT IDENTIFIER	Séquence d'entiers identifiant un objet à l'aide de son positionnement à partir de la racine de la MIB (ex. : 1.3.6.1)
ENUMERATION OF INTEGER	Permet de représenter une énumération de valeurs par des entiers non nuls (exemple : on(1), off(2))

Tableau 2.2 : Types simples d'ASN1

- **Les types construits :** le tableau suivant illustre les types construits utilisés par SNMP.

TYPE	DESCRIPTION
SEQUENCE	Liste ordonnée d'autres types ASN.1
SEQUENCE OF TYPE	Liste ordonnée d'éléments du même type

Tableau 2.3 : Types construits d'ASN1

6.1.3. Les macros

ASN.1 donne la possibilité de créer des macros. SNMP fait usage de macros pour avoir une information complète sur les objets de la MIB. Le langage ASN.1 définit cinq types de macros :

- **OBJECT-TYPE :** définit le nom d'objet de la MIB.
- **SYNTAX :** définit le type de l'objet selon la syntaxe ASN.1.
- **ACCESS :** indique le mode d'accès à l'objet (read-only, read-write,write-only, not-accessible).
- **STATUS :** indique si l'objet doit obligatoirement être présent dans toute implémentation de la MIB ou pas (valeurs possibles : mandatory,optional,obsolete).
- **DESCRIPTION :** un texte destiné à l'administrateur qui décrit l'objet.

Exemple

sysName **OBJECT-TYPE**
SYNTAX DisplayString (**SIZE** (0..255))
ACCESS read-write
STATUS mandatory
DESCRIPTION "An administratively-assigned name for this managed node. By convention, this is the node's fully-qualified domain name."

· **Syntaxe :** il s'agit d'une chaîne de caractères de taille variant entre 0 et 255.
· **Accès :** l'accès à cette variable se fait en lecture ou en écriture.
· **Etat :** cette variable existe et est toujours utilisable.
· **Description :** il s'agit du nom complet du nœud.
· **Place dans l'arborescence** : 5ème propriété de l'objet « system » : On en déduit que cette variable a pour OID la valeur 1.3.6.1.2.1.1.5

6.2. Types de SMI

SMI comporte des types simples, on cite parmi eux:

Type SMIv1	Type SMIv2	Description
IpAddress	IpAddress	Représente l'adresse IP 32 bits par un OCTET STRING de longueur 4.
Counter	Counter32	Représente un entier non nul dont la valeur croît jusqu'à sa valeur maximum puis repasse à zéro (ex : compteur de trames émises sur une interface). Représenté sur 32 bits.
Gauge	Gauge32	Jauge : entier positif dont la valeur peut croître ou diminuer (ex. : % de CPU).
TimeTicks		Permet de compter une durée en centièmes de seconde à partir d'un temps origine.
Opaque	(obsolète en v2)	Permettent de passer une syntaxe ASN.1 arbitraire sous forme d'un OCTET STRING.
	BITS	Ensemble de bits nommés.

Tableau 2. 4 : Types simples SMI pour les objets MIB

6.3. Types d'objets [OLW]

Il existe plusieurs types d'objets, parmi eux nous citons :

❖ **Les objets partiels** : qui servent à construire l'arborescence de la MIB. Ils n'ont pas de type, mais un OID.

Par exemple, l'objet system OBJECT IDENTIFIER ::= { mib-2 1 } groupe les objets de la mib-2 qui permettent d'identifier un agent SNMP.

❖ **Les objets scalaires** : Ce sont des objets qui ont une instance unique. Ils peuvent avoir un type simple (faisant partie de la syntaxe de base ASN.1), ou un type défini dans SMIv1 ou SMIv2.

Les instances des objets scalaires sont identifiées dans la MIB avec un OID

{ nomobjet .0 }, où nomobjet est la classe de l'objet.

Exemple

sysDescr.0, ou plus exactement .1.3.6.1.2.1.1.1.0 est l'instance de sysDescr qui est .1.3.6.1.2.1.1.1 (en ajoutant le suffixe .0 à .1.3.6.1.2.1.1.1).

❖ **Objets tabulaires** : une table est décrite comme une séquence de lignes qui sont elles-mêmes une séquence d'objets, ils sont décrits au moyen des types construits ASN.1.

7. Commandes SNMP [FFO]

Il existe trois types de commandes en SNMP, certains types possèdent des sous variantes:

- **GET**: Le manager émet une requête vers l'agent pour interroger une variable, celui−ci renvoi une réponse à cette commande;
- **GET−NEXT**: identique à GET, permet de parcourir la MIB.
- **SET**: Le manager émet une requête de modification sur un élément, l'agent répond par un *ACK*.
- **TRAP**: L'agent émet spontanément une information à destination d'une NMS.
- **GET−BULK (SNMPv2)**: GetBulk est une primitive qui mélange dans une même requête l'équivalent d'un Get et d'un GetNext, elle permet de faire des requêtes sur des "branches" complètes de l'arbre.

SNMPv2c a introduit quelques nouveaux types, mais sa nouveauté majeure est l'opération GETBULK, qui permet à une plate forme de gestion, de demander en bloc de plusieurs variables consécutives dans la MIB de l'agent.

- **INFORM (SNMPv2) :** Le message a le même format que GET et SET. Il permet à une station de gestion d'envoyer des informations vers une station d'administration qui centralise des informations contenues dans la MIB "manager-to-manager".

Une MIB "manager-to-manager" contient deux éléments :

- Un alarm group : permet de spécifier les paramètres de configuration des alarmes : intervalles entre les alarmes, instances ou objet ayant provoqué l'alarme.
- Un event group : permet de renseigner une station de gestion sur un ensemble d'événements choisis, sur l'instant où ils se produisent.

8. Description des paquets SNMP

Le transport de SNMP est basé sur UDP/IP, ce qui signifie que le message SNMP est encapsulé dans un datagramme UDP qui est lui-même contenu dans un datagramme IP.

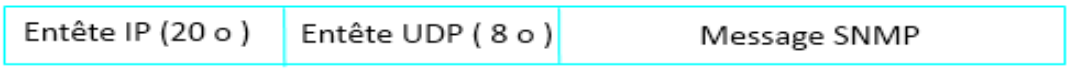

Figure 2.7 : Datagramme IP de SNMP

8.1. Format des messages [OLW]

Les messages SNMPv1 et v2c ont le même format :

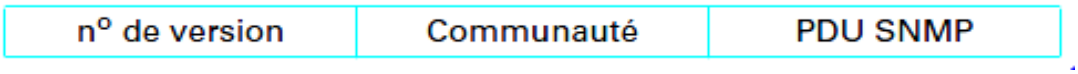

Figure 2.8 : Format d'un message SNMP

- **Le numéro de version** est 0 pour SNMPv1 et 1 pour SNMPv2.
- **Communauté** est un OCTET STRING qui représente un mot de passe à valider par l'entité SNMP.

8.2. Format des PDU [OLW]

Le format des **PDU** (Protocol Data Unit) **SNMP** contient les données du protocole de supervision. Il est construit de manière identique pour les requêtes et les réponses excepté les traps.

Figure 2.9 : Format d'un datagramme IP d'une requête SNMP (PDU)

- **Type :** est un octet identifiant la primitive (prend la valeur **0** pour GetRequest, **1** pour GetNextRequest, **2** pour GetResponse et **3** pour SetRequest).
- **Request ID** : est un entier identifiant la requête, qui permet de corréler commande et réponse (utilisé pour différencier les messages).

- **Le couple (*x*, *y*)** :

- Dans le cas de GET, SET, GetNext, SNMPv2-trap et Inform, la valeur du couple est (0,0).

- Dans le cas des *GetResponse*, *x* est le champ « error status » qui permet de retourner le code d'erreur concernant la requête correspondante, tandis que *y* est le champ « error index » qui permet d'indiquer la première variable dans la requête qui est en erreur.

Le champ « error status » prend la valeur 0 si aucune erreur ne s'est produite, sinon l'une des valeurs suivantes :

Réponse	**Description**
NoAccess	Accès non permis
WrongLenght	Erreur de longueur
WrongValue	Valeur erronée
WrongType	Type erroné
WrongEncoding	Erreur d'encodage
Nocreation	Objet non créé
ReadOnly	Lecture seule
NoWritable	Pas de permission d'écrire
AuthorisationError	Erreur d'autorisation

Tableau 2.5 : Valeurs de Error Status

- Pour GetBulk de SNMPv2, x représente la valeur « nonrepeater » qui indique le nombre de variables scalaires à récupérer, tandis que *y* représente la valeur de « maxrepetitions »qui est le nombre d'instances successives à récupérer pour les autres objets (équivalent du nombre de GetNext qu'il faudrait passer pour avoir le même résultat).

- **Variable bindings** : représente la liste des variables, avec pour chacune un OID et une valeur (ou un code d'erreur dans le cas de SNMPv2).

- **Format d'un Trap SNMP**

Figure 2.10 : Format d'un trap

- **Type** : prend la valeur 4.
- **Entreprise :** Type de l'objet gérant la notification.
- **Adresse agent :** Adresse de l'émetteur de la notification.
- **Trap générique :** indique l'identificateur du trap.

Un numéro de trap « generic-trap » permet d'identifier cinq traps « génériques » définis dans la mib-2, et de les séparer des traps « spécifiques » définis dans d'autres modules de MIB :

- **coldStart(0)** indique un reset complet de l'agent et probablement de l'équipement,
- **warmStart (1)** indique une réinitialisation,
- **linkDown (2)** indique le passage à l'état « down » d'une interface,
- **linkUp (3)** indique le passage à l'état « up » d'une interface,
- **authenticationFailure (4)** indique qu'une interrogation SNMP sur l'agent a été faite avec une mauvaise authentification,
- **egpNeighborLoss (5)** indique la perte d'un voisin EGP, quand ce protocole de routage est supporté par l'équipement,
- **(6)** indique qu'il s'agit d'un trap spécifique.

- **Trap spécifique :** une fois associé à l'OID, il permet d'identifier un événement spécifique.
- **Time-stamp :** indique la valeur du sysUp-Time (L'heure à laquelle le trap a été généré).
- **Variable bindings :** indique la liste des variables avec leur valeur au moment de la génération du trap.

9. Communauté SNMP [FFO]

Une communauté est une chaîne de caractères. Il existe par défaut deux types de communauté : **publique et privée.**

- Une communauté publique peut lire des informations, alors qu'une communauté privée peut lire et modifier des informations. On l'utilise dans les requêtes SNMP pour permettre d'établir des règles de sécurité. Ces règles ont pour but de restreindre l'accès aux utilisateurs aux différentes informations sur le réseau, et de vérifier et modérer les différents types d'actions auxquelles ils ont accès.

Un agent SNMP est finement paramétré pour répondre aux requêtes dont le paramètre communauté est validé dans sa configuration. Chaque communauté SNMP est définie par une chaîne d'octets, ainsi que le nom de la communauté.

- Afin d'assurer la sécurité des opérations, le protocole SNMP utilise la notion de message authentique. Il se définit comme un message pour lequel on a contrôlé que l'entité d'application qui l'émet est bien membre de la communauté spécifiée dans ce message.
- Une administration sécurisée utilisant des entités applicatives basées sur SNMP doit obligatoirement avoir des services d'authentification capables d'identifier et de contrôler la validité et l'authenticité des messages SNMP. La seule identification possible réside sur le paramètre « communauté ».

10. Transport dans SNMP [FFO]

SNMP est un protocole du modèle TCP/IP et peut donc être utilisé sur tous les réseaux de type Internet. Il exploite les capacités du protocole de transport UDP (User Datagram Protocol) :

- Chaque trame possède une adresse source et une adresse destination qui permet aux protocoles de niveaux supérieurs comme SNMP de pouvoir adresser leurs requêtes.
- Le protocole UDP peut utiliser un checksum optionnel qui couvre l'en-tête et les données de la trame. En cas d'erreur, la trame UDP reçue est ignorée : gage de fiabilité.
- Le protocole UDP fonctionne en mode non connecté, c'est-à-dire qu'il n'existe pas de lien persistant entre la station d'administration et l'agent administré. Cela oblige les deux parties à s'assurer que leurs messages sont bien arrivés à destination, ce qui apporte également un important gage de fiabilité pour la gestion réseau.
- Comme UDP n'assure pas la garantie de bonne délivrance des paquets, il est à la charge du manager SNMP de gérer des séquences de *timeout* et *retry* en cas de non réponse. **[OLW]**
- Deux ports sont désignés pour l'utilisation de SNMP :

- Port 161 pour les requêtes à un agent SNMP.
- Port 162 pour l'écoute des alarmes destinées à la station d'administration.

11. Fonctionnement de SNMP [FFO]

Lorsqu'une commande est expédiée à un agent, on attend de celui-ci une réponse. Plusieurs cas peuvent se produire :

- Aucune réponse (Temps d'attente dépassé).
- Erreur dans la requête.
- La requête a réussi.

Cas 1 : Aucune réponse

Plusieurs cas sont susceptibles de produire une absence de réponse de la part du matériel interrogé :

➢ Le fait que SNMP soit basé sur UDP n'assure pas le transfert des paquets à destination. Dans ce cas, le temps d'attente de réponse finit par s'écouler et il convient alors de réémettre la requête.

➢ Suivant l'implémentation des agents et la version de SNMP utilisée, si l'authentification échoue (mauvaise communauté, mot de passe incorrect), l'agent peut ne pas répondre à la requête.

➢ Le temps d'attente de réponse peut être paramétré dynamiquement et il est possible que le temps défini soit trop court pour permettre à la réponse de revenir.

➢ Dans le pire des cas, il est possible qu'il n'y ait pas d'agent SNMP disponible sur le matériel interrogé, donc pas de réponses à la requête.

- **Actions à entreprendre en cas de pertes de PDU :**

Une perte de PDU est envisageable et peut être la cause du mutisme de l'agent, voici quelques actions à entreprendre pour connaitre les raisons de ces pertes (et y remédier) :

- **Le paquet perdu est de type Get ou une réponse d'un agent administré**

En remarquant l'absence de réponse au GET émis, la station d'administration peut décider de renvoyer sa requête. L'identificateur de requête étant identique, il n'y a pas de risque de recevoir plusieurs réponses dues à la même requête.

En cas de non-réponse permanent, cette station peut déclarer ce périphérique muet comme étant défaillant.

- **Le paquet perdu est de type Set**

Il est alors plus judicieux d'envoyer un GET sur cette entrée, afin de savoir si c'est la requête qui a été perdue ou sa réponse.

- **La perte d'un trap**

Comme aucun acquittement n'est associé à un trap, la station d'administration ne sera jamais avertie par le signal. Il est donc nécessaire que cette dernière scrute régulièrement l'état de ses agents pour s'informer des éventuels changements pour lesquels aucune alarme n'a été reçue.

Cas 2 : Erreur

Plusieurs cas sont susceptibles de conduire au renvoi d'une erreur :

- Lors d'une écriture sur une variable en lecture seule.
- Lorsque la valeur définie d'une variable est de type incorrect (écriture d'une chaîne de caractères dans une variable de type entier par exemple).
- Lorsque la variable n'existe pas.
- Lorsque la trame SNMP est incorrecte (corruption, longueur non valide…).
- Lorsque l'authentification SNMPv3 a échoué.

Cas 3 : Réussite

- Lorsque la requête à l'agent SNMP réussit, celui-ci envoie la valeur de la variable accédée (que ce soit en lecture ou en écriture).

12. Le protocole SNMPv3

Les deux versions de SNMP ont rencontré plusieurs lacunes en ce qui concerne la sécurité. C'est pourquoi, il a fallu mettre en œuvre un mécanisme permettant de sécuriser les transactions, et ainsi offrir une gestion plus sûre à l'administrateur.

12.1. Architecture de SNMPv3

L'architecture de SNMPv1 et SNMPv2 est basée sur un modèle agent/manager. La structure du cadre d'administration a été redéfinie par la suite, en posant l'essentiel du SMIv2 ainsi que l'enrichissement du dialogue entre entités SNMP. SNMPv3 reprend donc, en les englobant, SNMPv1 et SNMPv2.

L'entité SNMPv3 [SVH]

Dans SNMPv3, le concept agent/manager s'efface au profit de la notion d'entité SNMP (« snmp entity »), qui selon le cas aura un rôle de manager, d'agent ou les deux à la fois, en employant la terminologie SNMPv1.
Une entité SNMP est constituée d'un moteur (*engine*) et d'applications.

La figure suivante illustre l'entité SNMPv3, l'application MIB Instrumentation concerne l'agent. En réalité, les deux entités Agent et Manager sont identiques, cela dit certaines applications diffèrent selon la fonction de chacun d'entre eux.

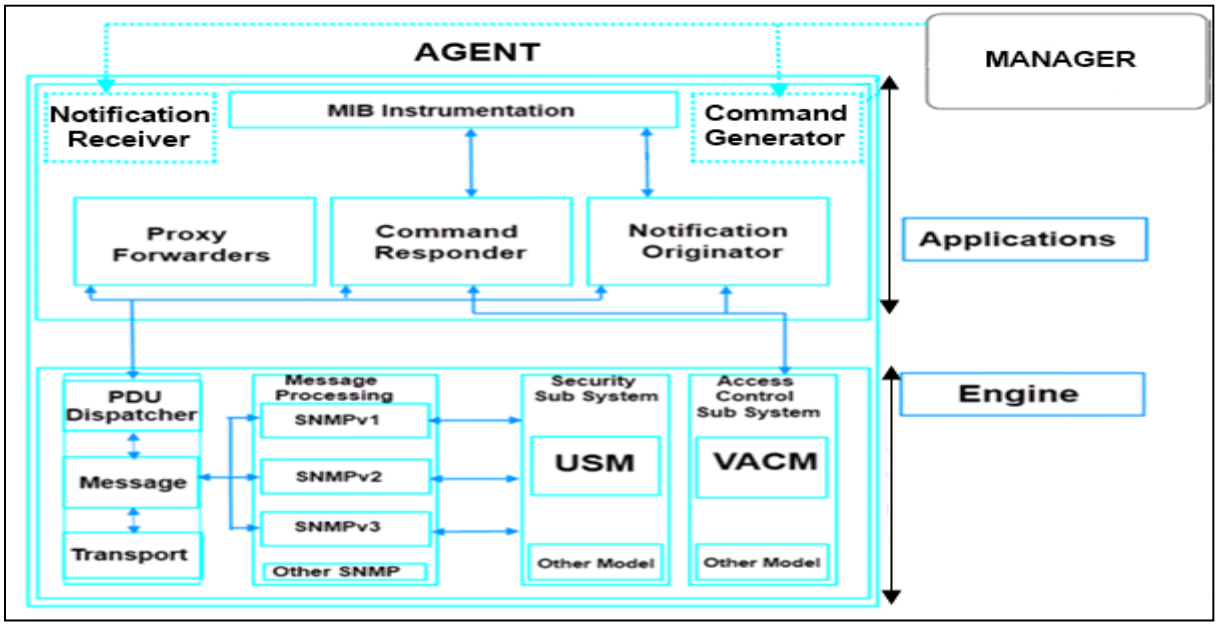

Figure 2.11 : L'entité SNMP [SVH]

- **Engine :** machine (ou moteur) SNMP, chargée de l'instrumentation du protocole.

Comme son nom l'indique, le moteur assure toute la partie technique du travail. Il contient :

- Le *dispatcher :* oriente les messages qui arrivent du réseau vers le bloc de traitement de messages qui sera capable de les traiter. Réciproquement, il redirige les messages en provenance de ce module vers le réseau, encapsulé dans le protocole adéquat.
- Le *message processing :* s'occupe du décodage et de l'assemblage des messages. Des sous-systèmes gèrent les différents types de messages utilisables par l'entité (SNMPv1, SNMPv2C, SNMPv3).
- Le module *security :* traite la sécurité des échanges ; la confidentialité des communications ainsi que l'authentification du correspondant. Plusieurs protocoles sont définis et utilisables et l'architecture SNMPv3 est extensible de manière à permettre l'utilisation d'autres modèles de sécurité que ceux prévus d'origine (USM User-based Security Model).
- Le module *access control :* traite quant à lui le contrôle d'accès aux ressources de la contrôlée par l'entité SNMP, en fonction des prérogatives accordées. Un modèle est défini dans SNMPv3 : VACM (View-based Access Control Model) qui permet de définir des contrôles d'accès sur la MIB.

- **Applications :**

- *Command Generator :* désigne la fonction Manager en SNMPv1, il émet les requêtes.
- *Command Responder :* désigne l'ancienne fonction agent, il reçoit les requêtes générées par le Command Generator et y répond.
- Notification Originerator : génère les notifications de l'agent (regroupant TRAP et INFORM.
- Notification Responder : reçoit les notifications générées par le Notification Generator, elle est propre au Manager.
- Proxy-forwarder : permet de traduire un message d'une version SNMP dans une autre version.

Les applications s'appuient sur ce *moteur* SNMP pour échanger les unités de données de protocole (PDU).

La structure modulaire permet de faire coexister les applications agent (Command Responder) et gestionnaire (Command Generator). Un gestionnaire peut se comporter en agent vis à vis d'un autre gestionnaire.

12.2. Format des paquets SNMPv3 [MAI]

Le paquet SNMPv3 suit le format suivant

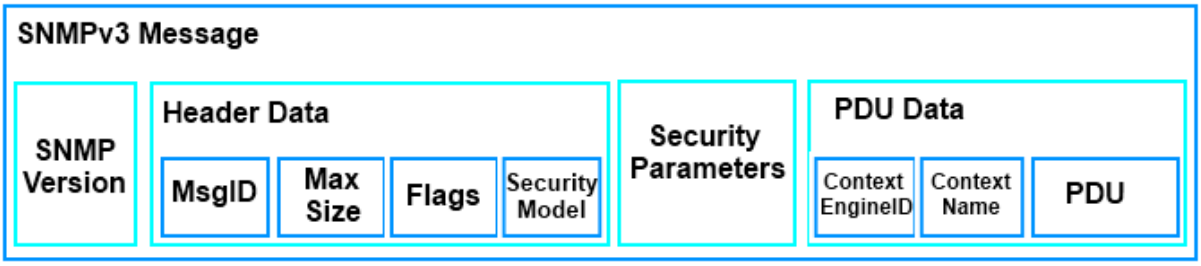

Figure 2.12 : Description du paquet SNMPv3

- **Version SNMP :** Pour SNMPv3, on place la valeur 3 dans ce champ.
- **Identificateur de message (MsgId):** Utilisé pour associer les requêtes et les réponses, on retrouve souvent des algorithmes, où le premier message de requête est envoyé avec un nombre aléatoire et les suivants avec les incréments de 1. Les paquets qui sont émis en réponse à une requête portent la même identification que le paquet de la requête.
- **Grandeur maximale du paquet réponse (Max Size):** Le moteur choisit la taille maximale d'une réponse à une requête selon ses capacités en mémoire tampon et ses limites à décoder de longs paquets. Quand on envoie une réponse à une requête, on doit veiller à ne pas dépasser la taille maximale.
- **Drapeaux(Flags) :** indiquant si une réponse est attendue et si un modèle de sécurité a été utilisé.

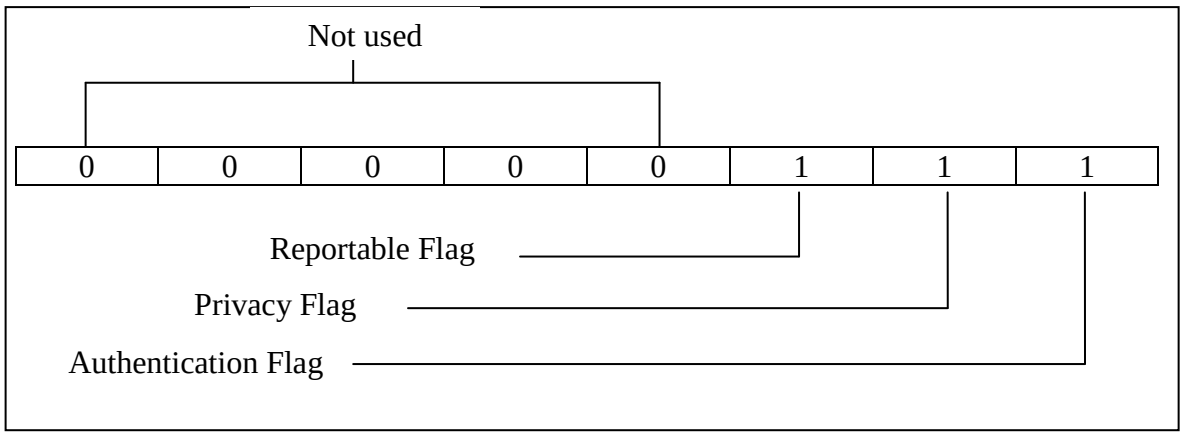

Figure 2.13 : Les drapeaux SNMPv3 [MAI]

Trois bits sont utilisés (sur les 8) pour indiquer :

- Si une réponse est attendue à la réception de ce paquet. (Reportable Flag).
- Si un modèle de privatisation a été utilisé (Privacy Flag).
- Si un modèle d'authentification a été utilisé (Authentication Flag).

Il n'est pas possible d'envoyer un paquet qui utilise un modèle de privatisation sans qu'il soit préalablement authentifié, il faudrait donc qu'il utilise un modèle d'authentification.

- **Le modèle de sécurité utilisé :** Ce module identifie le type de sécurité qui est utilisé pour crypter le reste du paquet. Le groupe de l'IETF recommande qu'il y ait peu de modules de sécurité pour permettre de maximiser l'inter-compatibilité des équipements.
L'algorithme de privatisation DES (Data Encryption Standard) et l'algorithme d'authentification MD5 ont été choisis comme algorithmes utilisés dans SNMPv3. L'algorithme SHA est optionnel.

- **Informations de sécurité :** Comme les clés publiques, les arrangements entre protocoles de sécurité…

- **Identifie le contexte :** à l'OID se rajoute le contexte qui permet d'avoir plusieurs instances de MIB sur une entité SNMP. Par exemple, pour un équipement implémentant deux ponts, le contextName permettra d'identifier la MIB du pont d'où on désire interroger.

- **PDU :** Les valeurs demandées ou les réponses à des demandes.

13. Sécurité de SNMP

L'une des plus grandes faiblesses du protocole SNMP dans ses deux premières versions est l'absence d'un mécanisme adéquat pour assurer la confidentialité et la sécurité des fonctions de gestion.

En effet, la sécurité reposait sur la connaissance mutuelle (entre agent et gestionnaire) d'une chaîne de caractères (la communauté) insérée dans le message. Cette communauté n'étant pas cryptée, elle est interceptable sur le réseau.

- **Sécurité apportée par SNMPv3 [SDR]**

Cette nouvelle version du protocole SNMP vise essentiellement à inclure la sécurité des transactions. La sécurité comprend l'identification des parties qui communiquent et l'assurance que la conversation soit privée, même si elle passe par un réseau public.
Cette sécurité est basée sur 2 concepts :

- **USM** (User-based Security Model)
- **VACM** (View- based Access Control Model)

13.1. User Security Module (USM)

Trois mécanismes sont utilisés. Chacun de ces mécanismes a pour but d'empêcher un type d'attaque.

- **Authentification et intégrité :** L'intégrité a pour rôle d'assurer que le paquet reste inchangé pendant la transmission tandis que l'authentification garantit l'identité de l'émetteur.

Pour construire ce mécanisme, on doit avoir connaissance des fonctions de hachage à une seule direction. Des exemples de ces fonctions sont : MD5 et SHA. Ces fonctions 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, 20 octets pour SHA).

Pour authentifier l'information qui va être transmise, on doit avoir un mot de passe partagé. Le mot de passe ne doit donc être connu que par les deux entités qui s'envoient les messages, et préférablement par personne d'autre. La figure ci-dessous illustre le mécanisme d'authentification :

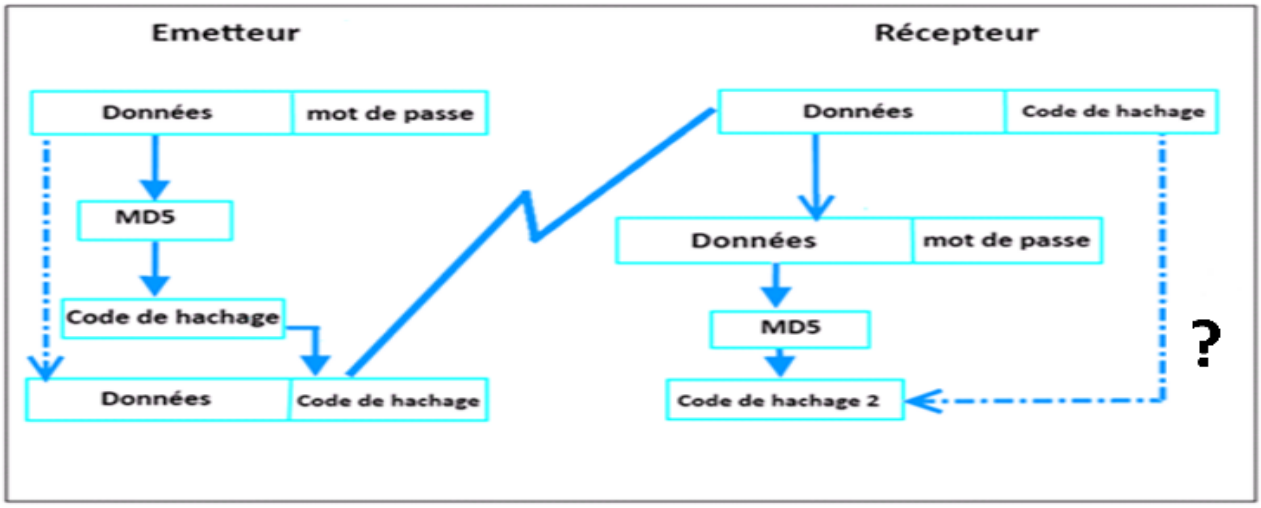

Figure 2.14 : Authentification avec SNMPv3 [SDR]

Les étapes d'authentification sont les suivantes :

- Le transmetteur groupe des informations à transmettre avec le mot de passe.
- Le groupe passe ensuite dans la fonction de hachage à une direction.
- Les données et le code de hachage sont ensuite transmis sur le réseau.
- Le receveur prend le bloc des données, et y ajoute le mot de passe.
- Le groupe passe dans la fonction de hachage à une direction.
- Si le code de hachage est identique à celui transmis, le transmetteur est authentifié.

Avec cette technique, le mot de passe est validé sans qu'il ait été transmis sur le réseau. Quelqu'un qui saisit les paquets SNMPv3 passants sur le réseau ne peut pas trouver le mot de passe facilement.

Le résultat de la fonction de hachage est placé dans le bloc paramètres de sécurité du paquet SNMPv3.

L'authentification se fait sur tout le paquet SNMPv3, elle ne vise pas à cacher l'existence du paquet ou à le crypter. Si l'authentification et l'intégrité sont utilisées, les espions qui saisissent les paquets passants sur le réseau peuvent encore en voir le contenu. Toutefois, ils ne peuvent pas changer le contenu sans connaître le mot de passe ce qui permet de préserver l'intégrité du message.

Privatisation d'opérations

Afin d'effectuer des communications sécurisées, SNMPv3 se base sur DES (Data Encryption Standard) puisqu'il permet de crypter les messages SNMP pour assurer la privatisation des opérations.

Le cryptage a pour but d'empêcher qu'un intrus n'obtienne les informations de gestion contenues dans un paquet SNMPv3 en écoutant les requêtes et les réponses qui circulent sur le réseau.

Avec SNMPv3, le cryptage de base se fait sur un mot de passe « partagé » entre le manager et l'agent. Ce mot de passe ne doit être connu par personne d'autre. Pour des raisons de sécurité, deux mots de passe sont utilisés : un pour l'authentification et un autre pour le cryptage (privatisation). Ceci permet au système d'authentification et au système de cryptage d'être indépendants et de ne pas se compromettre.

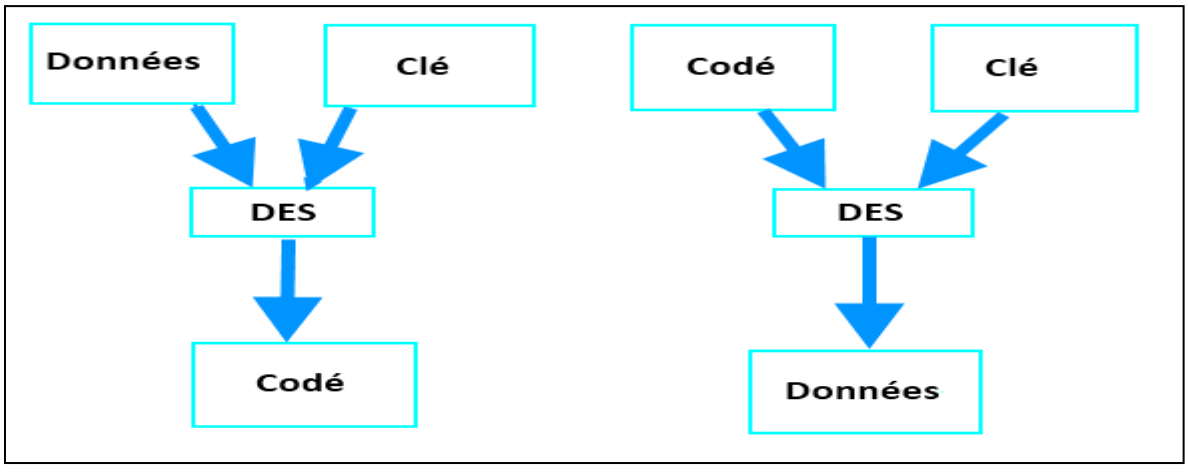

Figure 2.15 : Le cryptage DES [SDR]

- **Principe de DES**

On utilise une clé de 64 bits (8 des 64 bits sont des parités, la clé réelle est donc de 56 bits) et DES encrypte 64 bits à la fois. Comme les informations que l'on doit crypter sont plus longues que 8 octets, on utilise le chaînage de blocs DES de 64 bits.

Une combinaison du mot de passe, d'une chaîne aléatoire et d'autres informations forme le « Vecteur d'initialisation ». Chacun des blocs de 64 bits passe par DES et est chaîné avec le bloc précédent avec un XOR. Le premier bloc est chaîné par un XOR au vecteur d'initialisation. Le vecteur d'initialisation est transmis avec chaque paquet dans le « Paramètres de sécurité », un champ qui fait partie du paquet SNMPv3.

Contrairement à l'authentification qui est appliquée à tout le paquet, le cryptage est seulement appliqué sur le PDU.

La figure ci-dessous montre le fonctionnement du cryptage pour le chaînage des blocs DES :

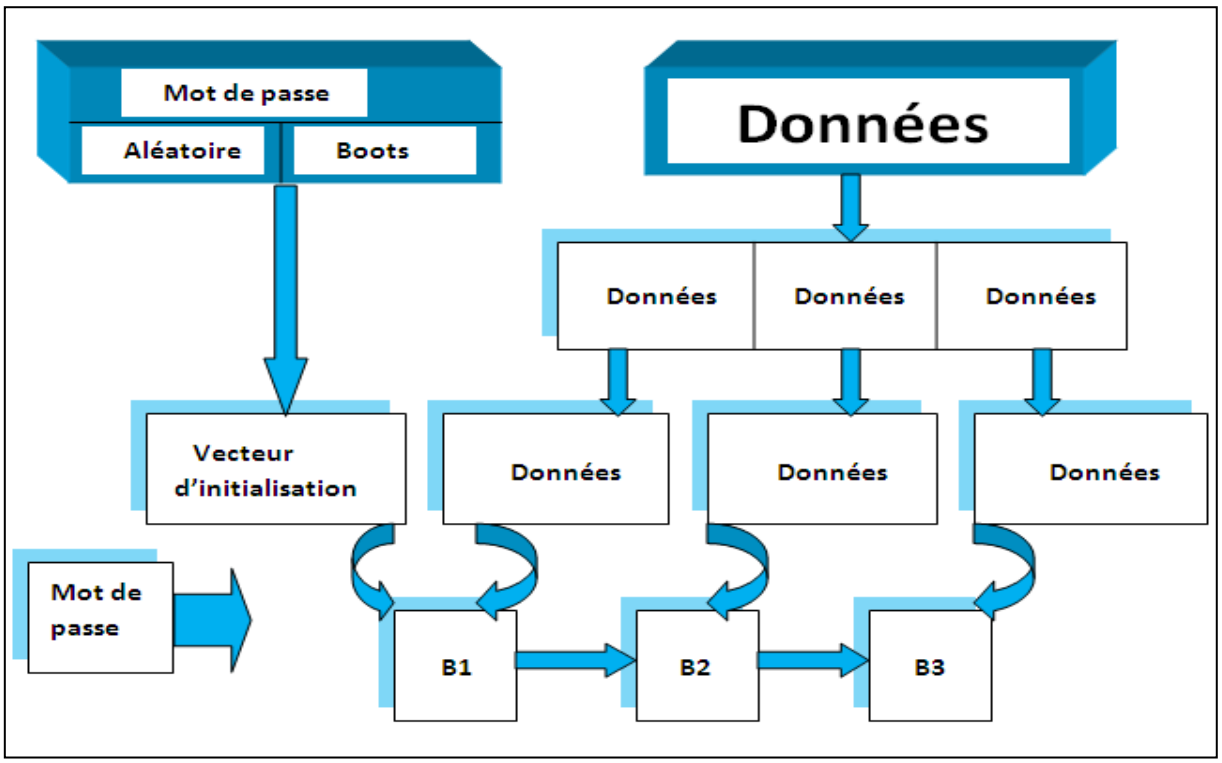

Figure 2.16 : Fonctionnement du cryptage pour le chaînage DES [MAI]

La RFC 3826 relate l'intégration d'AES dans SNMP. Celui-ci renforce le cryptage en remplaçant DES.

- **L'estampillage du temps** : Empêche la réutilisation d'un paquet SNMPv3 valide déjà transmis par un manager ; si une requête est transmise, les mécanismes d'authentification, et de cryptage n'empêchent pas quelqu'un de saisir un paquet SNMPv3 valide du réseau et de tenter de le réutiliser plus tard, sans modification.

Par exemple, si l'administrateur effectue l'opération de mise à jour d'un équipement; une tierce personne peut saisir ce paquet et tenter de le retransmettre à l'équipement à chaque fois que celle-ci désire faire une mise à jour illicite de l'équipement. Même si la elle n'a pas l'autorisation nécessaire, elle envoie un paquet, authentifié et crypté correctement par l'administration à l'équipement.

On appelle cette attaque le « replay attack ». Pour éviter ceci, le temps est estampillé sur chaque paquet. Quand on reçoit un paquet SNMPv3, on compare le temps actuel avec le temps dans le paquet. Si la différence est plus que supérieur à 150 secondes, le paquet est ignoré.

Toutefois, les agents SNMP sont souvent des micro-contrôleurs qui n'ont pas à leur disposition une horloge avec l'heure courante et précise. Les horloges doivent être mises à l'heure, et il est pratique de les garder à l'heure même quand l'équipement est éteint. De plus, il est difficile de garder toutes les horloges synchronisées.

SNMPv3 n'utilise pas l'heure normale, on utilise plutôt une horloge différente dans chaque agent qui garde le nombre de secondes depuis que l'agent a été mis en fonctionnement, ce compteur est appelé TIME, on garde aussi un compteur, appelé BOOTS qui compte le nombre de fois où l'équipement a été mis en fonctionnement.
La combinaison du BOOTS et du 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 plate-forme de gestion doit garder une horloge qui doit être synchronisée pour chaque agent qu'elle contacte. Au moment du contact initial, la plate-forme obtient la valeur du BOOTS/TIME de l'agent et synchronise une horloge distincte **[MAI].**

13.2. View Access Control Model (VACM)

Permet de définir des vues sur la MIB, vues sur lesquelles les opérations SNMP pourront être limitées. Par exemple, il sera possible d'accéder à des tables de configuration en lecture mais pas en écriture, ou encore il ne sera pas possible du tout d'accéder, même en lecture, à certaines parties de la MIB **[OLW].**

14. Conclusion

Nous avons abordé dans ce chapitre le protocole SNMP son architecture, ses différentes requêtes et leurs formats ainsi que la limitation des deux premières versions et l'arrivée de la troisième qui résout le problème majeur de la sécurité.

Pour modéliser notre application, nous avons opté pour le langage UML afin d'exprimer les besoins et les fonctionnalités attendues de l'application.

Chapitre 3

Modélisation de l’application

1. Introduction

Après une étude approfondie du protocole SNMP, nous allons nous baser essentiellement sur la version 3 pour réaliser l'application.

Dans ce chapitre, nous procédons à la modélisation de l'application de gestion réseau avec le langage UML (Unified Modeling Language). Comme UML n'impose pas de démarche de modélisation, nous avons suivi un procédé centré sur l'utilisateur et guidé par les cas d'utilisation.

2. Présentation du langage UML

2.1. Historique

UML (Unified Modeling Language), traduit par langage de modélisation objet unifié, est un langage dont l'apparition est à l'origine de l'union de trois méthodes qui ont le plus fait leurs preuves dans le milieu de la modélisation objet aux années 1990, et qui sont OMT (Object Modeling Technique), OOSE (Object Oriented Software Engineering) et booch. Il a ensuite été normalisé par l'OMG (Object Management Group) en 1997.

2.2. Définition du langage UML

UML est un langage de modélisation fondé sur les concepts objets. Il supporte un riche ensemble d'éléments de notation graphique très spécifique ainsi que les règles grammaticales s'y attachant pour élaborer des modèles de logiciels, il décrit la notation pour les classes, les composants, les nœuds, les activités, les cas d'utilisation, les objets et les états, ainsi que la façon de modéliser les relations entre ces éléments. Le but d'UML est d'unifier les langages de modélisation afin que tous puissent utiliser les mêmes concepts, en faisant abstraction de leur implémentation, la raison pour laquelle UML n'est pas restreint à un domaine d'application spécifique **[MLA].**

UML aide à construire des modèles rigoureux, maintenables et aptes à être suivis, supportant le cycle de vie complet du développement **[UCA].**

Le langage UML.2 fournit 13 diagrammes qui sont classés selon deux types :

- Diagrammes structurels : Ces diagrammes ont vocation à représenter l'aspect statique d'un système. La vue structurelle du modèle UML est la vue la plus utilisée pour spécifier une application. L'objectif de cette vue est de modéliser la structure des différentes classes d'une application orientée objet ainsi que leurs relations. L'aspect structurel comprend les diagrammes suivants :

- Diagramme de classes.
- Diagramme d'objets.
- Diagramme de composants.
- Diagramme de déploiement.
- Diagramme de paquetage.
- Diagramme de structures composites.

- Diagrammes comportementaux : Les diagrammes comportementaux sont focalisés sur la description de la partie dynamique du système à modéliser. L'aspect dynamique d'une application est défini par la façon dont interagissent les objets qui composent l'application. À l'exécution, l'objet est l'entité de base d'une application. Les objets qui composent une application pendant son exécution et leurs échanges de messages permettent à l'application de réaliser les traitements pour lesquelles elle a été développée. Sept diagrammes sont proposés :

- Diagramme de cas d'utilisation.
- Diagramme d'activités.
- Diagramme d'états-transitions.
- Diagramme de séquence.
- Diagramme de communication.
- Diagramme global d'interaction.
- Diagramme de temps.

3. Phase1 : Analyse des besoins

L'analyse des besoins est une étape primordiale dans la modélisation. Les fonctionnalités de l'application désirée doivent être établies en fonction des besoins recensés. L'application doit assurer les services suivants :

- Interrogation des nœuds du réseau en utilisant les requêtes SNMP,
- Gestion des MIB,
- Gestion des notifications,
- Représentation des données sous plusieurs formats : tableaux ou graphes.

Pour la modélisation de notre application, nous avons opté pour les diagrammes suivants :

- Diagramme de cas d'utilisation.
- Diagramme de séquence.
- Diagramme d'activité.

3.1. Diagramme de cas d'utilisation : est un graphe qui définit le système, les acteurs, les cas d'utilisations ainsi que les liens entre acteurs et cas d'utilisations.

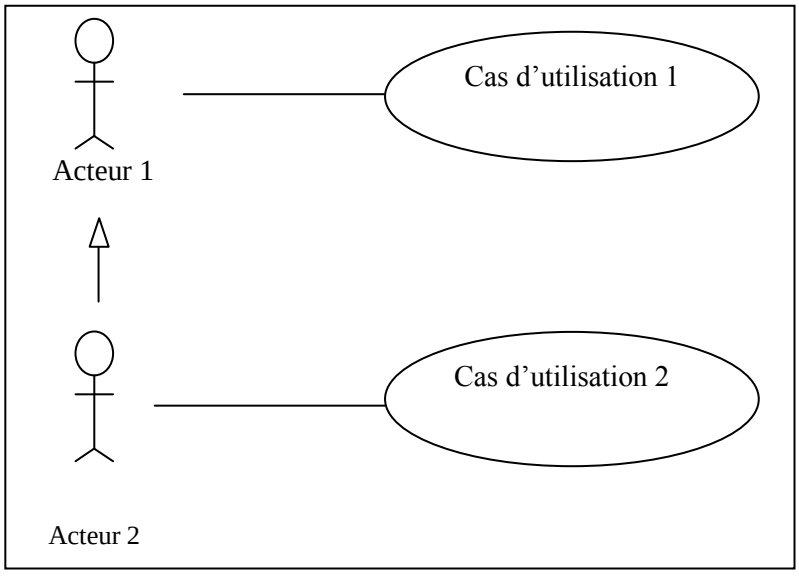

Figure 3.1 : Formalisme de représentation du diagramme de cas d'utilisation

- **Diagramme de cas d'utilisation de l'application**

a. Identification des acteurs

Nous distinguons les deux acteurs suivants dans l'application :

- L'administrateur : qui a pour rôle de gérer de la MIB.
- L'agent : qui a pour rôle de maintenir la MIB et d'envoyer des traps à l'administrateur.

b. Identification des cas d'utilisation

Cas d'utilisation	**Acteur**	**Actions**
Gestion de la MIB	*Administrateur*	Chargement, déchargement et consultation de la MIB.
Requêtes SNMP	*Administrateur*	Requêtes qui permettent de gérer la MIB.
Gestion de traps reçus	*Administrateur*	Ecoute des traps émis par l'agent.
Requêtes sécurisées	*Administrateur*	Requêtes de mise à jour ou consultation avec authentification et /ou privatisation.
Requêtes non sécurisées	*Administrateur*	Requêtes de mise à jour ou consultation avec authentification par communauté.
Authentification par communauté	*Administrateur*	Accès à la MIB par un nom de communauté de lecture ou d'écriture.
Authentification privée	*Administrateur*	Avec protocoles et mots de passe d'authentification privatisation.
Générer des traps	*Agent*	Envoi de notifications à l'administrateur.
Exécuter les requêtes	*Agent*	Exécution des requêtes émises par l'administrateur.
Maintenir la MIB	*Agent*	Mise à jour et consultation des variables de la MIB.

Tableau 3.1 : Identification des cas d'utilisation

c. Diagramme de cas d'utilisation de l'application

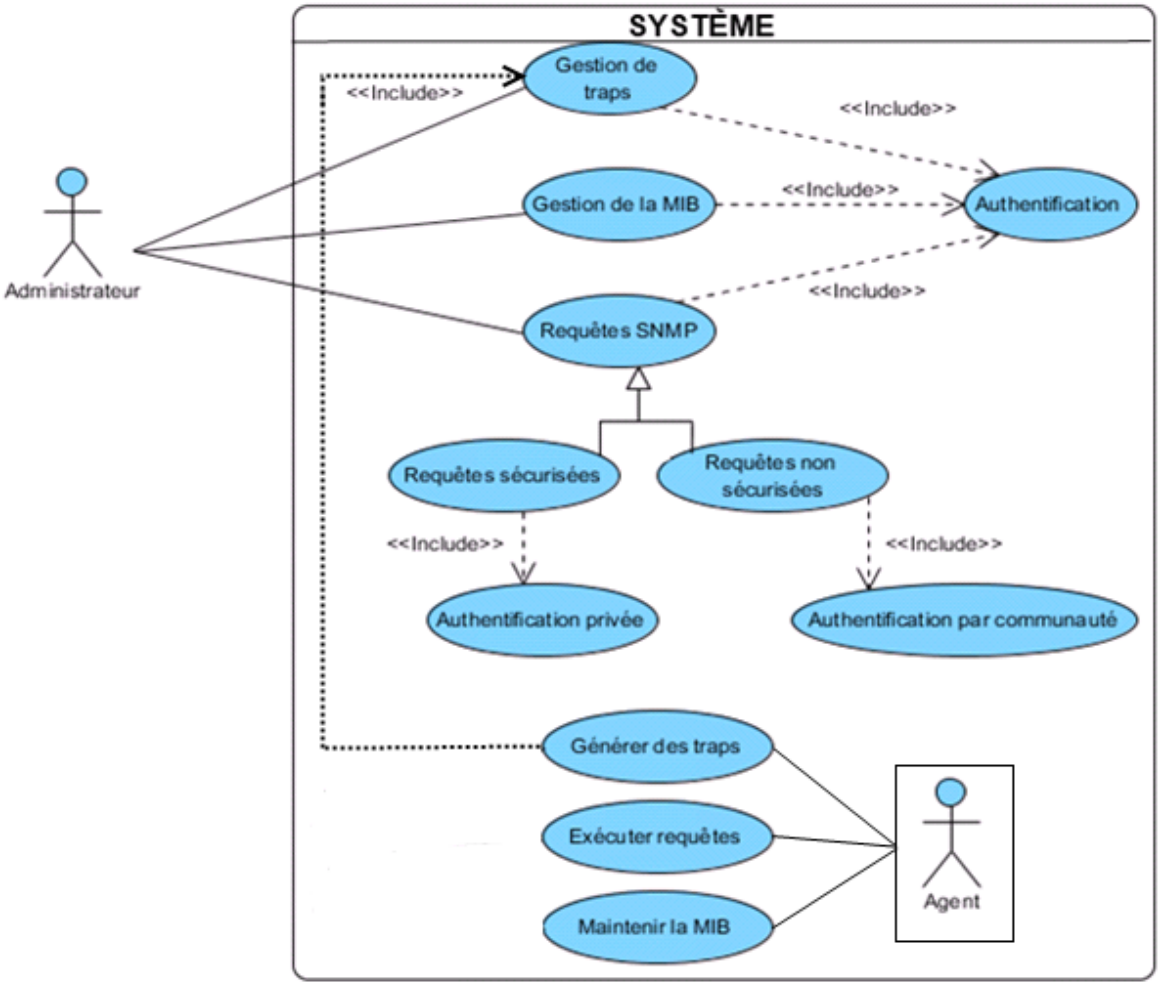

Figure 3.2 : Diagramme de cas d'utilisation

d. Description des cas d'utilisation

Cas d'utilisation	Authentification
Titre **But** **Acteurs**	Authentification. Donner accès à l'application. Administrateur.
Description des enchainements	
Pré condition **Enchainement** **Enchainement 2**	Exécuter l'application. Saisie du login et du mot de passe. Vérification du login et mot de passe dans la table des utilisateurs.

Tableau 3.2 : Description du cas d'utilisation « Authentification »

Cas d'utilisation	Gestion de traps
Titre	Gestion de traps.
But	S'informer des évènements qui ont eu lieu dans le réseau.
Acteurs	Administrateur.
Description des enchainements	
Pré condition	Administrateur authentifié.
Enchainement	L'agent émet un trap.
Enchainement 2	L'administrateur traite le trap reçu.

Tableau 3.3 : Description du cas d'utilisation « Gestion de traps »

Cas d'utilisation	Gestion de la MIB
Titre	Gestion de la MIB.
But	Charger et décharger la MIB pour consultation ou autre.
Acteurs	Administrateur.
Description des enchainements	
Pré condition	Administrateur authentifié.
Enchainement	L'administrateur charge ou décharge une MIB.

Tableau 3.4: Description du cas d'utilisation « Gestion de la MIB »

Cas d'utilisation	Requêtes SNMP
Titre	Requêtes SNMP.
But	Mise à jour ou consultation des objets de la MIB.
Acteurs	Administrateur.
Description des enchainements	
Pré condition	Administrateur authentifié
Enchainement	Envoi d'une requête SNMP à l'agent.
Enchainement 2	L'agent reçoit la requête et l'exécute en fonction de son type.

Tableau 3.5 : Description du cas d'utilisation « Requêtes SNMP »

Cas d'utilisation	Authentification privée
Titre	Authentification privée
But	Empêcher les intrus de s'approprier les commandes envoyées sur le réseau.
Acteurs	Administrateur.
Description des enchainements	
Pré condition	Administrateur authentifié.
Enchainement	Envoi d'une requête SNMP à l'agent et saisie du protocole d'authentification (privatisation) ainsi que des mots de passe respectifs.
Enchainement 2	L'agent reçoit la requête et la traite en fonction de son type.

Tableau 3.6: Description du cas d'utilisation « Authentification privée »

Cas d'utilisation	Exécuter requêtes
Titre	Exécuter requêtes.
But	Mise à jour ou consultation des objets de la MIB.
Acteurs	Agent.
Description des enchainements	
Pré condition	Réception de requêtes émises par l'administrateur.
Enchainement	Exécution de la requête.
Enchainement 2	Renvoie du résultat à la station d'administration.

Tableau 3.7 : Description du cas d'utilisation « Exécuter requêtes »

Cas d'utilisation	Maintenir la MIB
Titre	Maintenir la MIB.
But	Mise à jour ou consultation des objets de la MIB.
Acteurs	Agent.
Description des enchainements	
Pré condition	Réception d'une requête d'un administrateur.
Enchainement	Exécution de la requête.
Enchainement 2	Mettre à jour la MIB.

Tableau 3.8 : Description du cas d'utilisation « Maintenir la MIB »

Cas d'utilisation	Générer des traps
Titre **But** **Acteurs**	Générer des traps . Notifier des événements à l'administrateur. Agent.
Description des enchainements	
Pré condition **Enchainement** **Enchainement 2**	Ecouteur de traps lancé par l'administrateur. Ecouter et générer des traps Envoyer les notifications à l'administrateur.

Tableau 3.9 : Description du cas d'utilisation « Générer des traps »

3.2. Diagramme de séquence : montre les interactions entre les objets, agencées en séquence dans le temps ; il montre en particulier les objets participant à l'interaction par leurs lignes de vie et les messages qu'ils s'échangent ordonnancés dans le temps. Cependant, il ne montre pas les relations entre les objets.

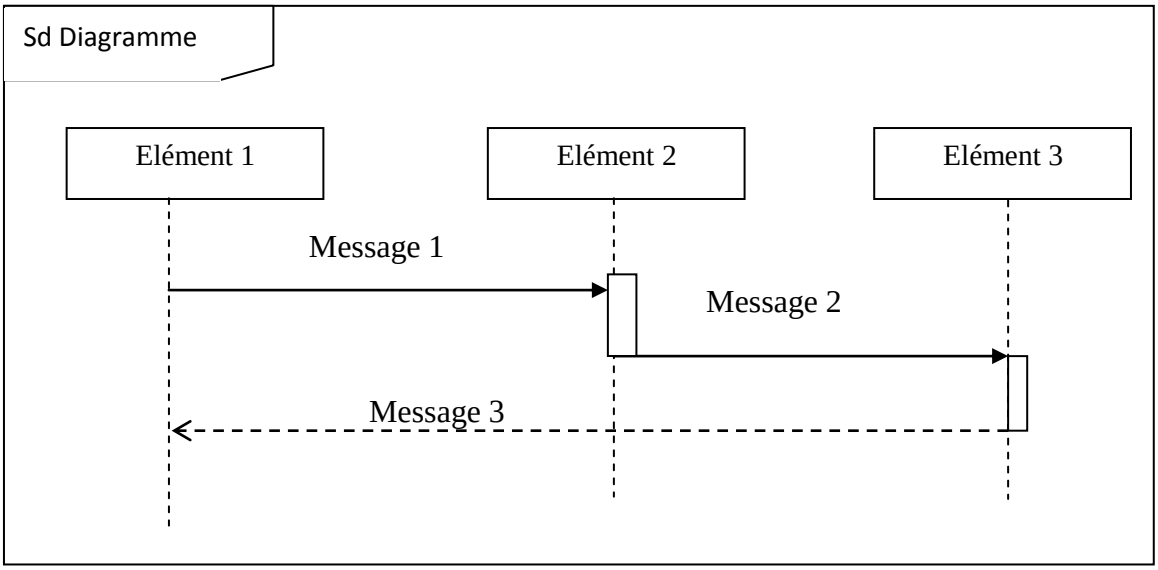

Figure 3.3 : Formalisme de représentation du diagramme de séquence

- **Diagramme de séquence du cas d'utilisation « Authentification à l'application » :**

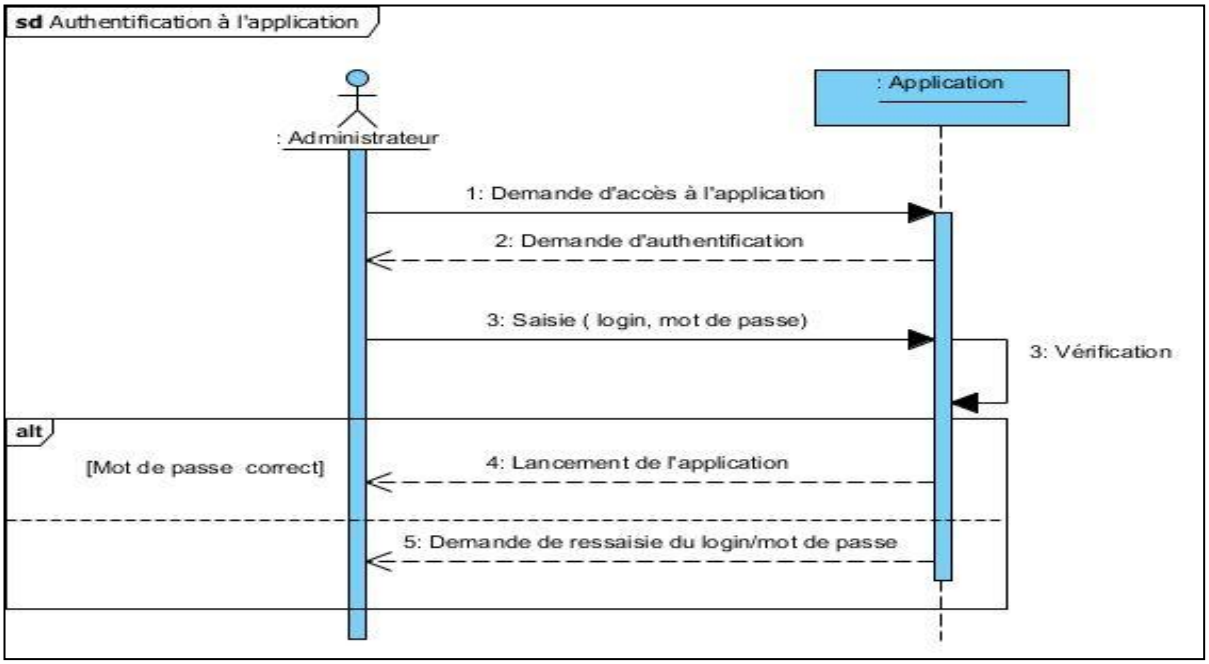

Figure 3.4 : Authentification à l'application

- **Diagramme de séquence du cas d'utilisation « Chargement d'une MIB »**

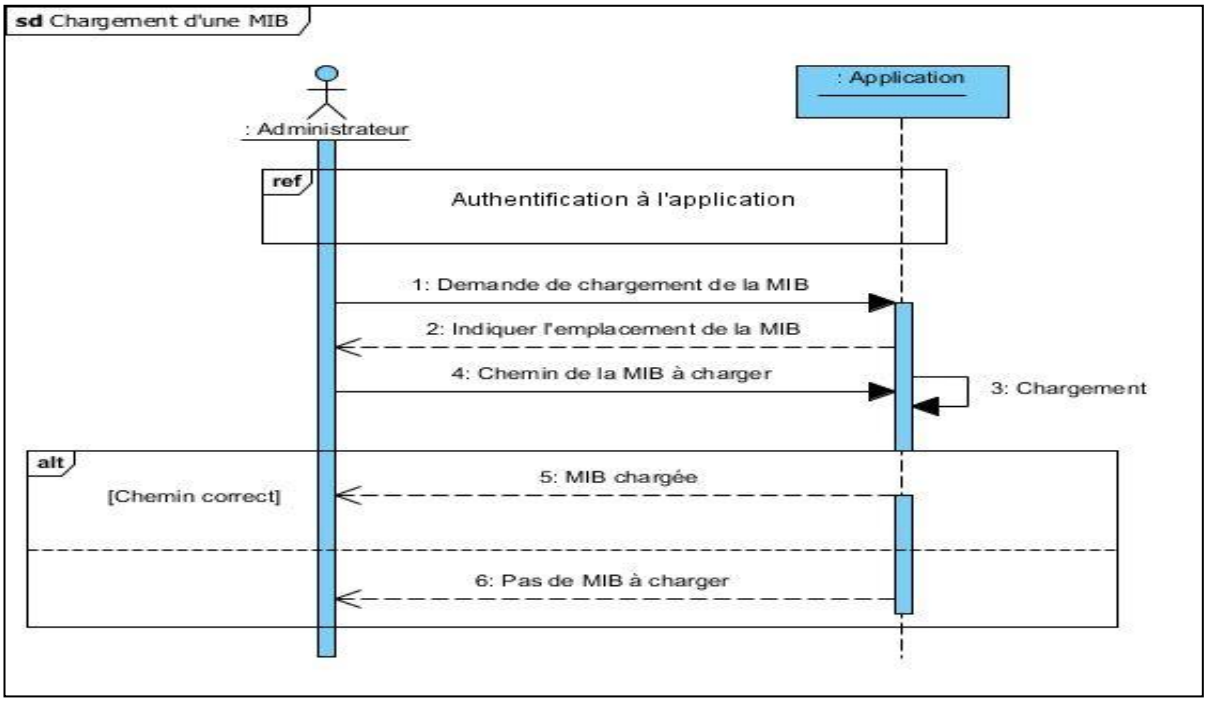

Figure 3.5 : Chargement d'une MIB

- **Diagramme de séquence du cas d'utilisation « Déchargement d'une MIB »**

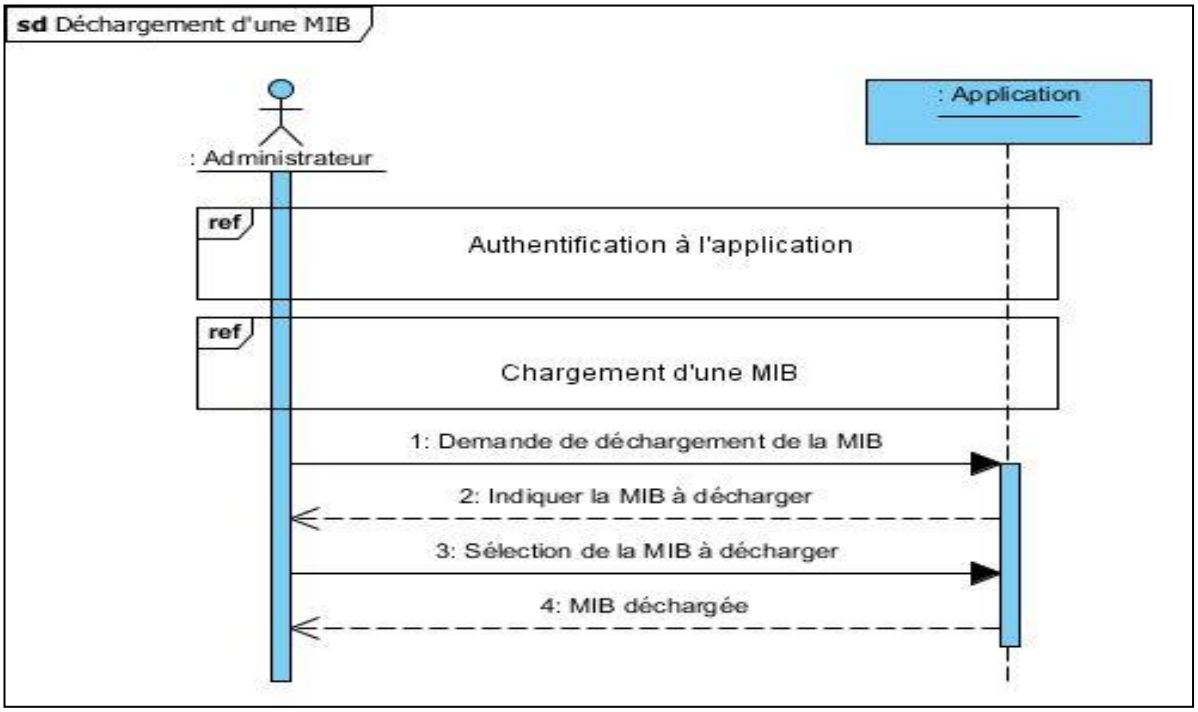

Figure 3.6 : Déchargement d'une MIB

- **Diagramme de séquence du cas d'utilisation « Sélectionner un objet de la MIB »**

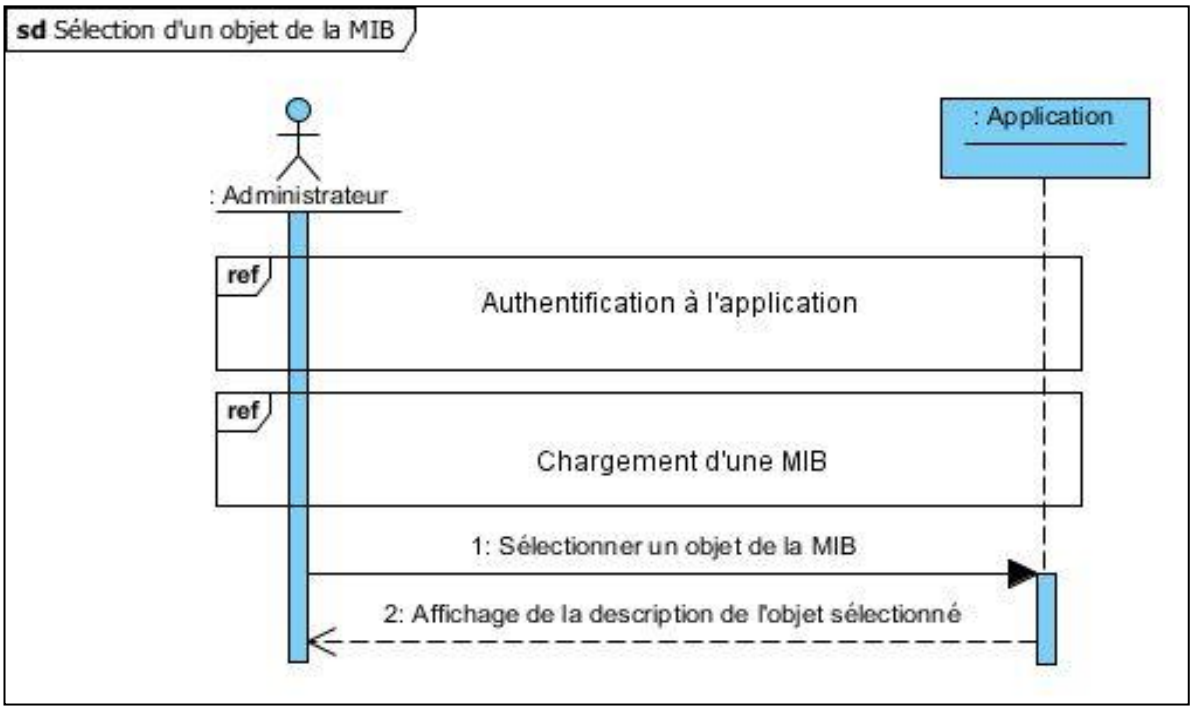

Figure 3.7 : Sélection d'un objet de la MIB

- **Diagramme de séquence du cas d'utilisation « Envoyer une requête »**

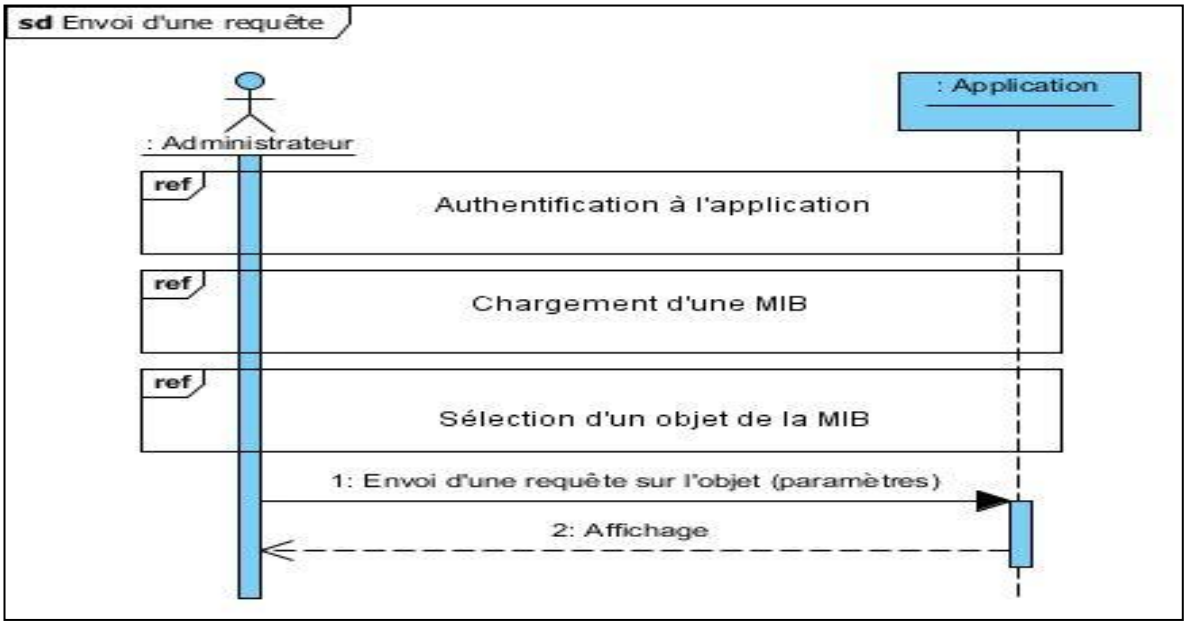

Figure 3.8 : Envoyer une requête

- **Diagrammes de séquence du cas d'utilisation « Requête SNMP »**
  - **Diagrammes de séquence de requêtes non sécurisées**

❖ **GET**

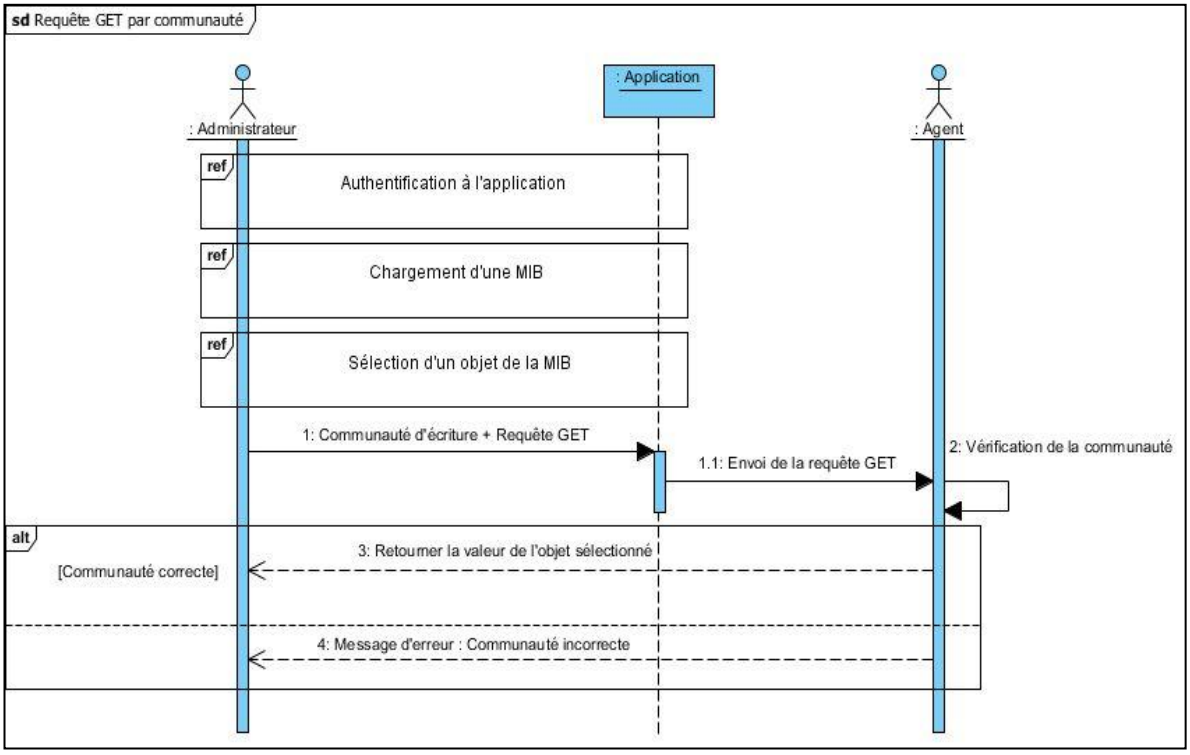

Figure 3.9 : Requête GET

❖ **GetNext**

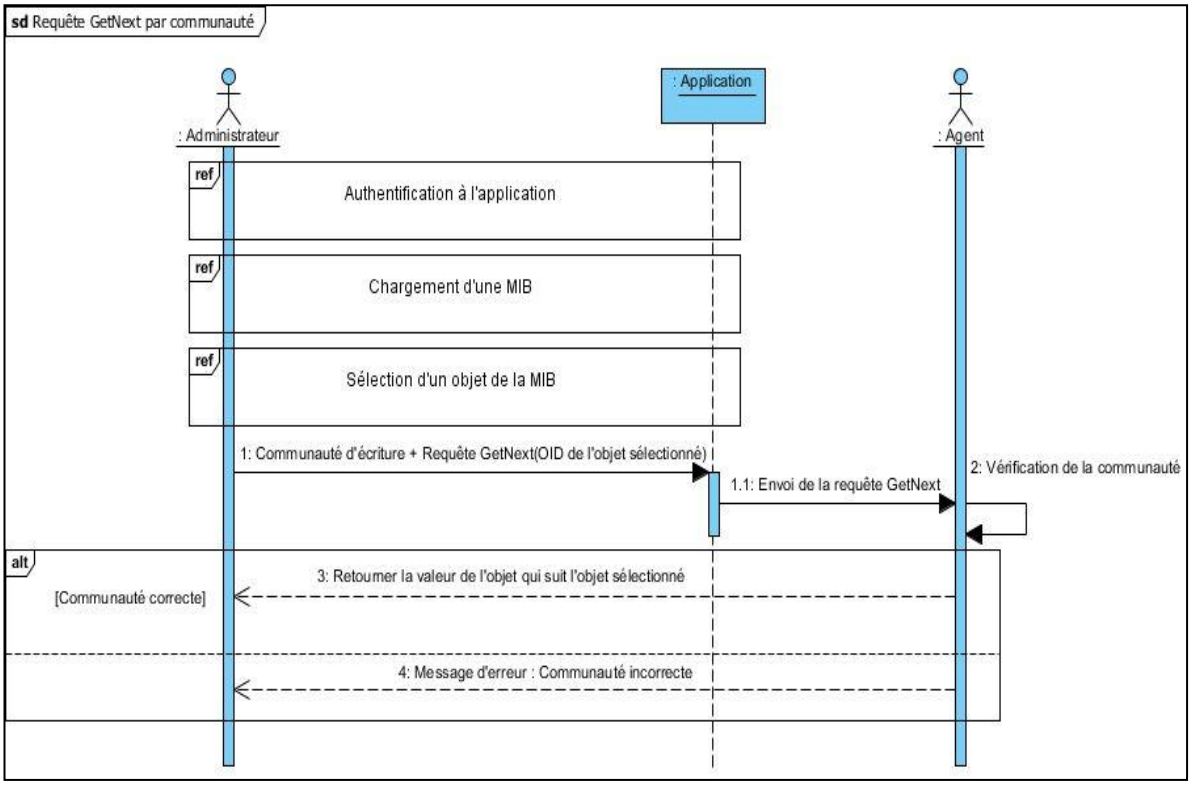

Figure 3.10 : Requête GetNext

❖ **GetBulk**

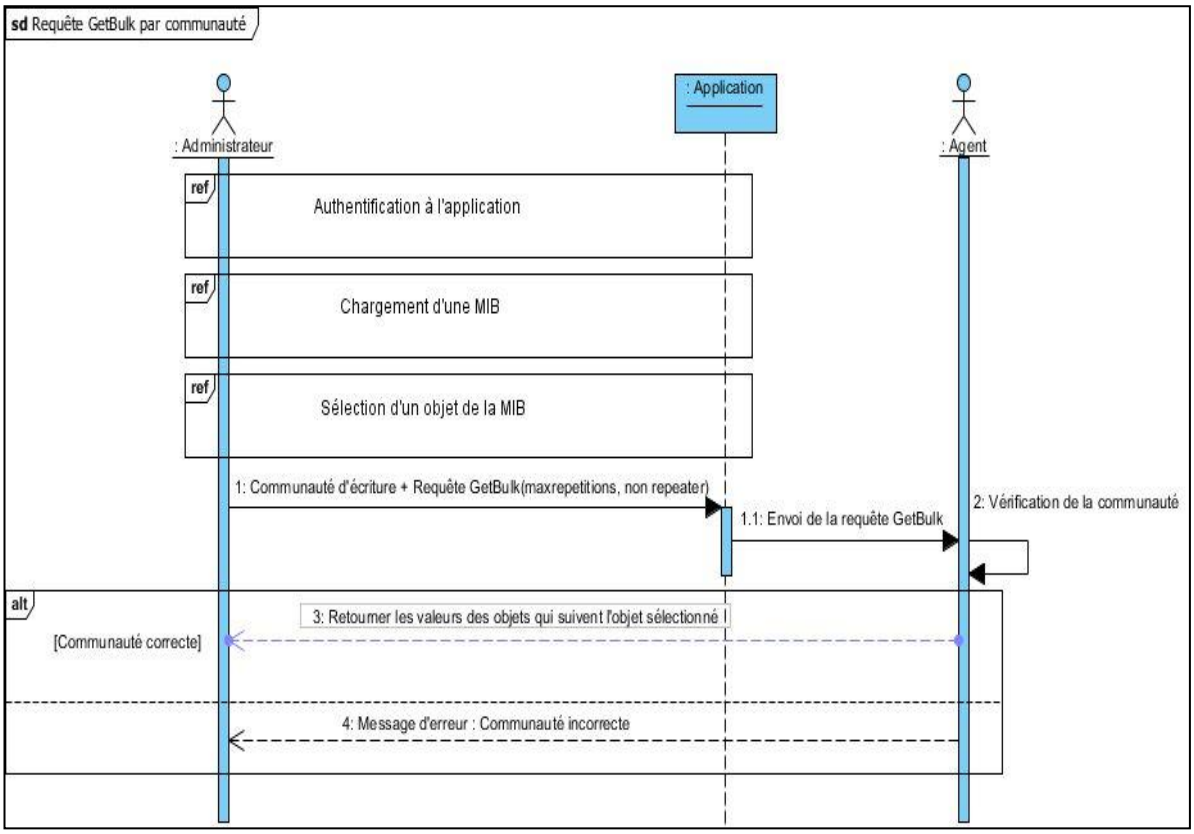

Figure 3.11 : Requête GetBulk

❖ **SET**

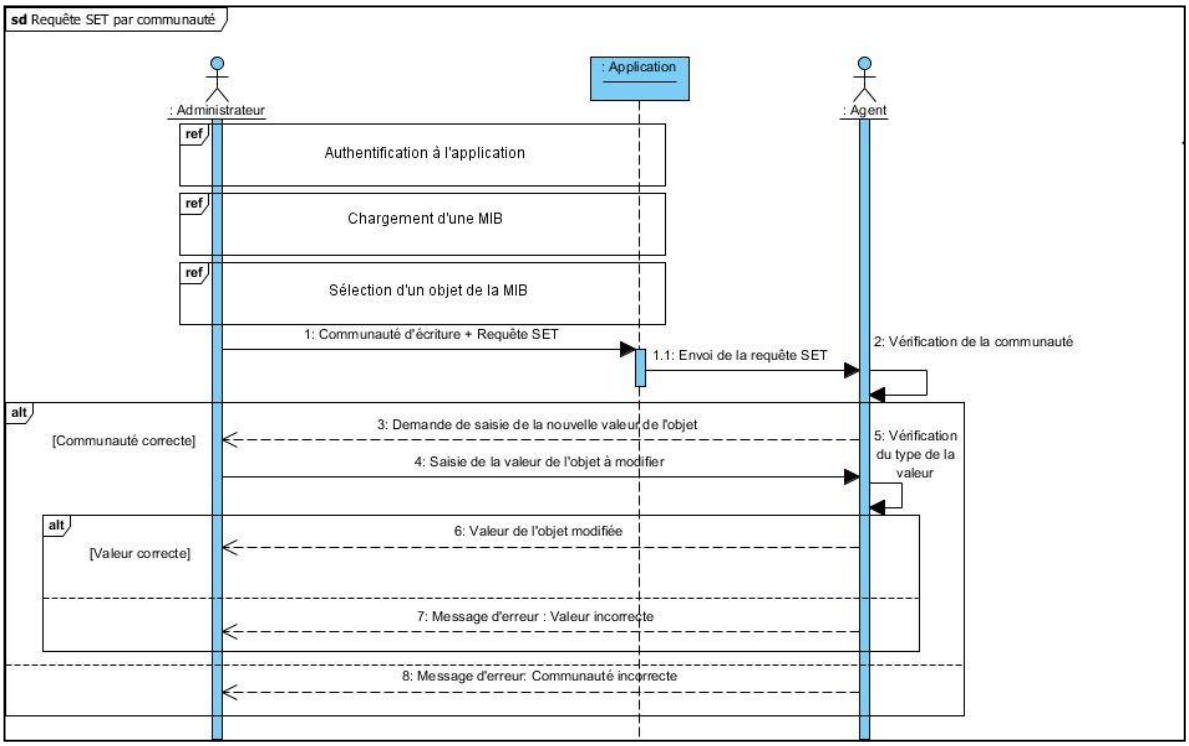

Figure 3.12 : Requête SET

- **Diagrammes de séquence de requêtes sécurisées**

❖ **GET**

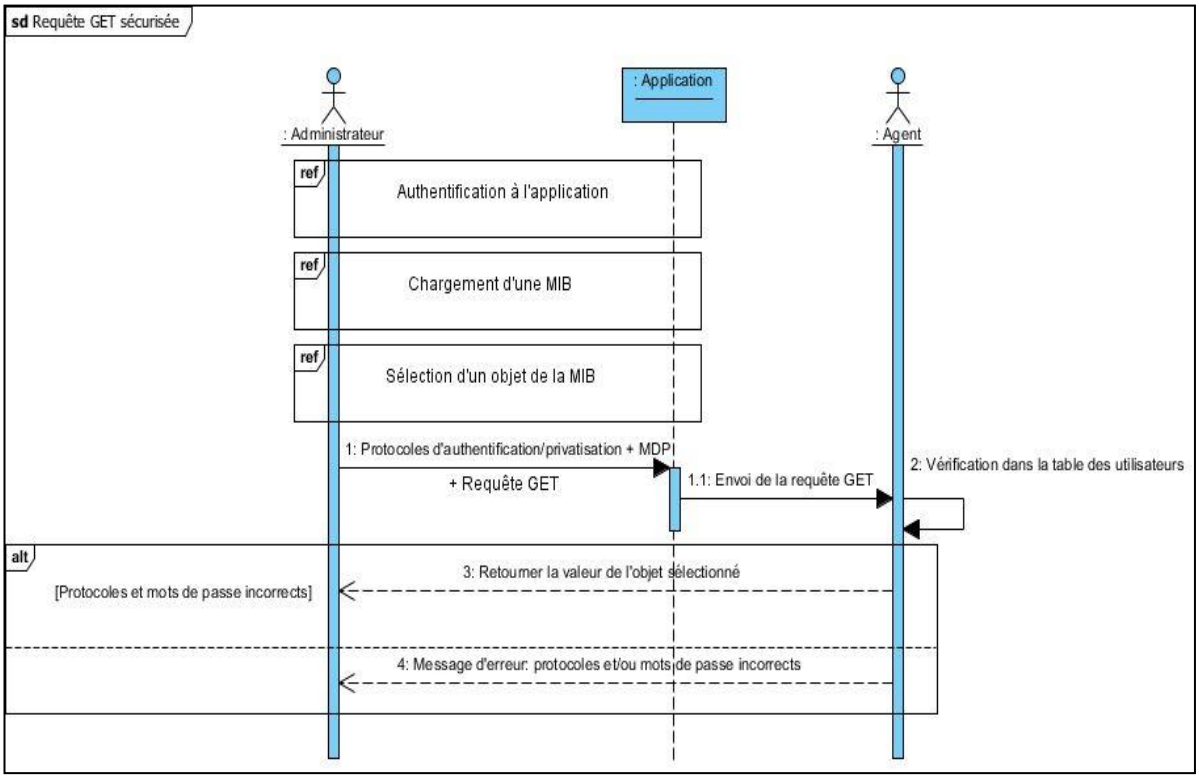

Figure 3.13 : Requête GET sécurisée

❖ **GetNext**

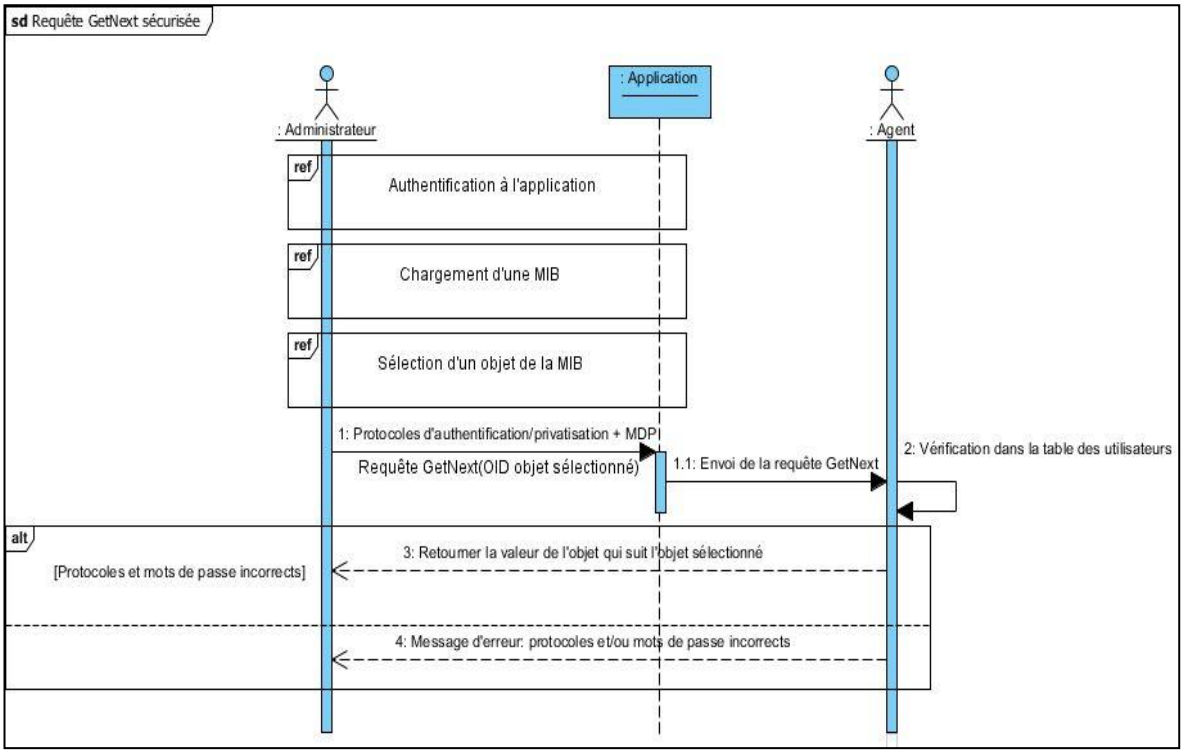

Figure 3.14 : Requête GetNext sécurisée

❖ **GetBulk**

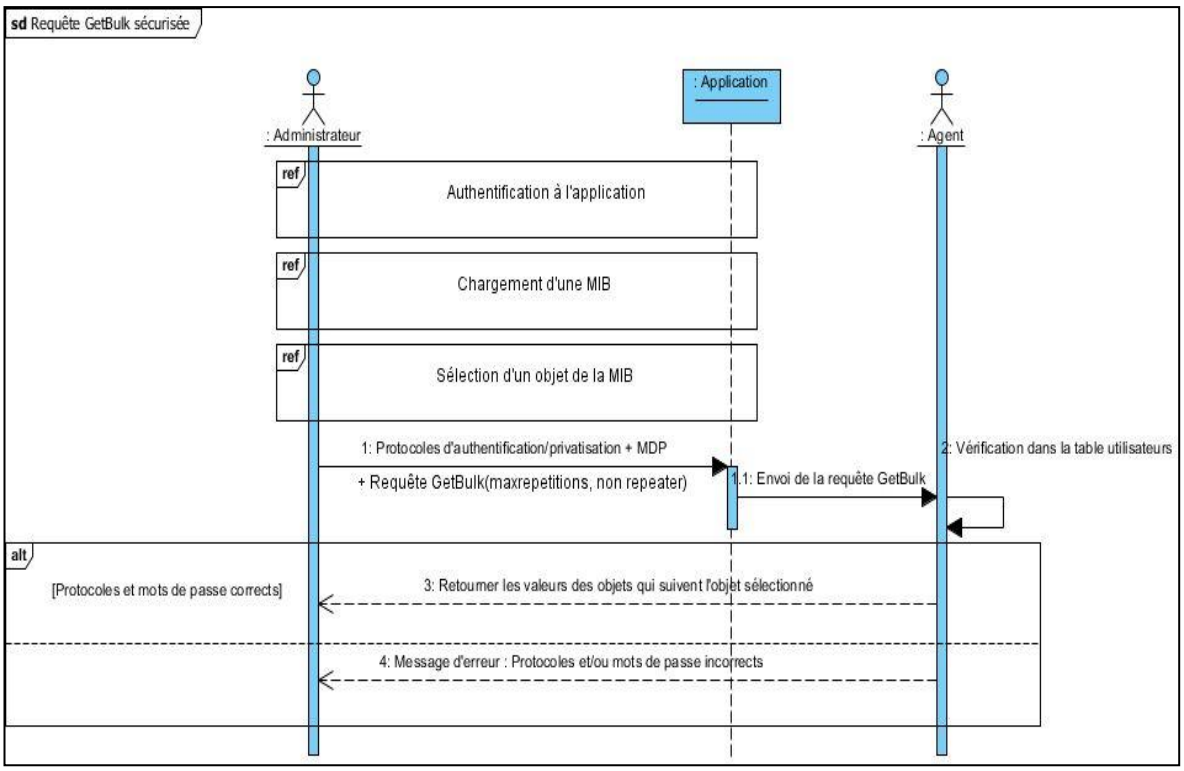

Figure 3.15 : Requête GetBulk sécurisée

❖ **SET**

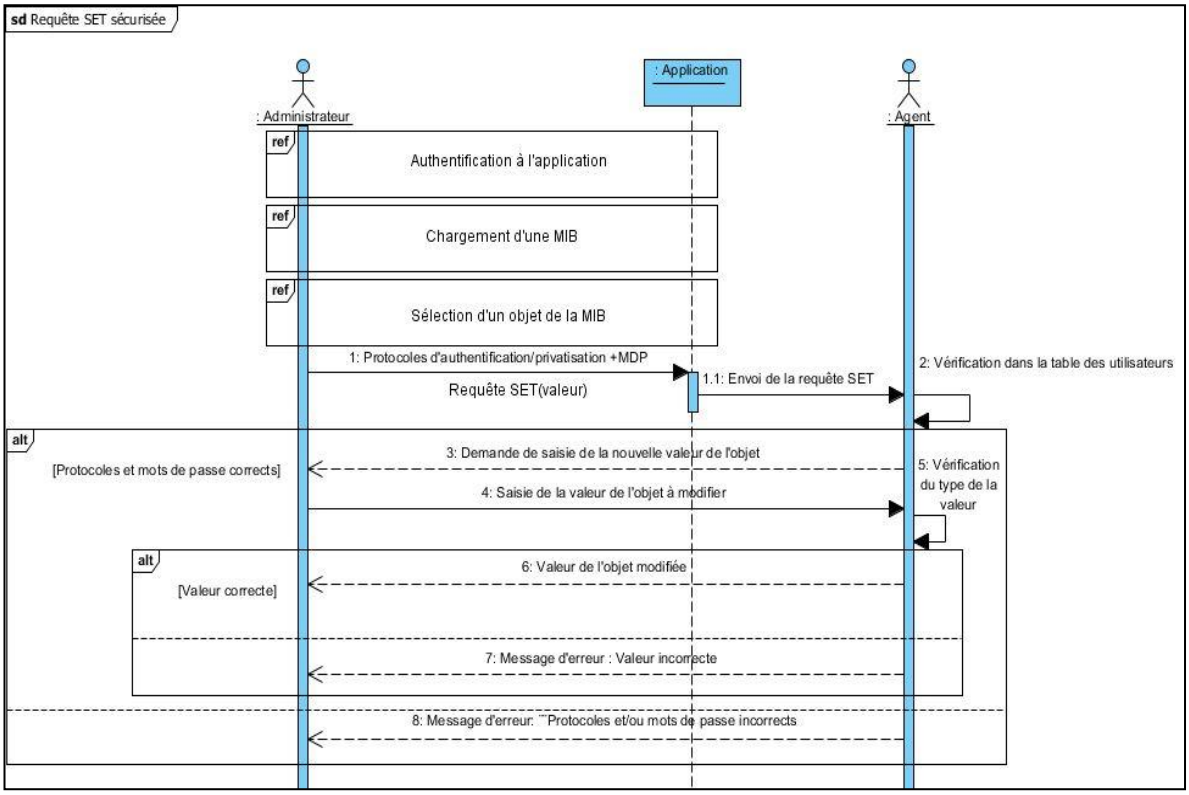

Figure 3.16 : Requête SET sécurisée

- **Diagramme de séquence du cas d'utilisation « Gestion de traps »**

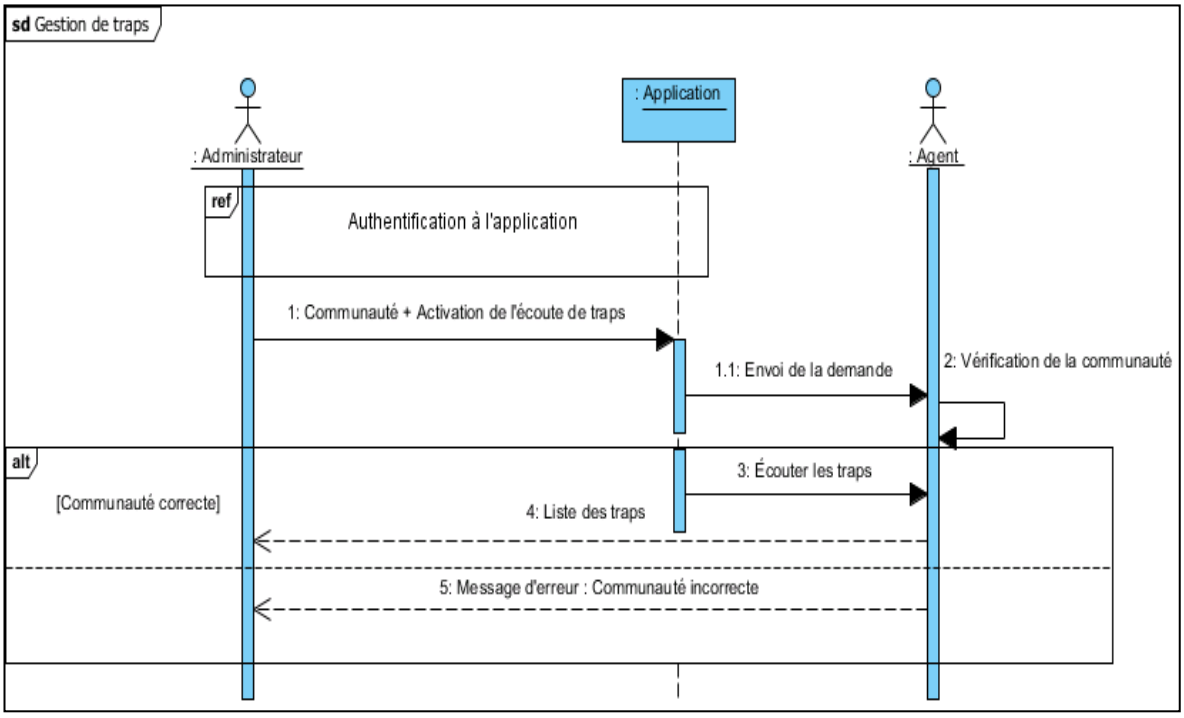

Figure 3.17 : Gestion de traps

3.3. Diagramme d'activité : représente une machine à états particulière dans laquelle les états sont des activités représentant la réalisation d'opérations et où les transitions sont déclenchées par la fin des opérations.

Un diagramme d'activité est attaché dans sa globalité à une classe ou à l'implémentation d'une opération ou encore à un cas d'utilisation.

Un état action est un état qui contient une action interne et au moins une transition sortante.

Un état action peut déclencher plusieurs transitions qui doivent être accompagnées chacune d'une condition de garde toutes mutuellement exclusives (concept de décision).

Un diagramme d'activité peut être découpé en couloirs représentant chacun une responsabilité de l'activité et assignée à un objet (concept de partition).

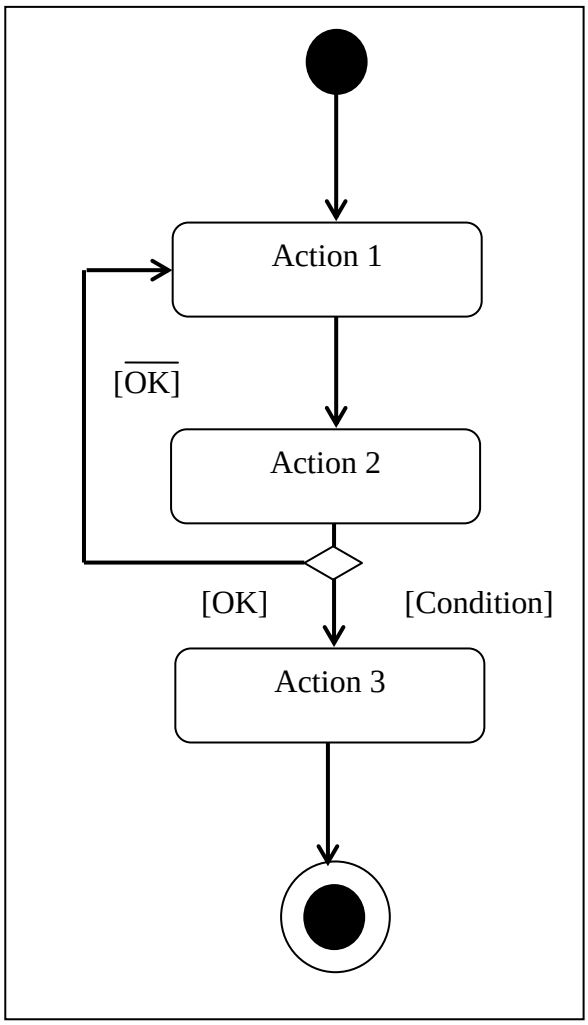

Figure 3.18 : Formalisme de représentation du diagramme d'activité

Afin d'accéder à l'application, l'utilisateur doit d'abord s'authentifier, après cela, il charge une MIB en indiquant son emplacement.

Une fois celle-ci affichée, il sélectionne une variable de l'arborescence afin d'effectuer une requête qui peut être sécurisée ou pas dessus.

Dans le cas de requêtes sécurisées, il lui faudra indiquer les protocoles d'authentification et privatisation ainsi que leurs mots de passes correspondants.

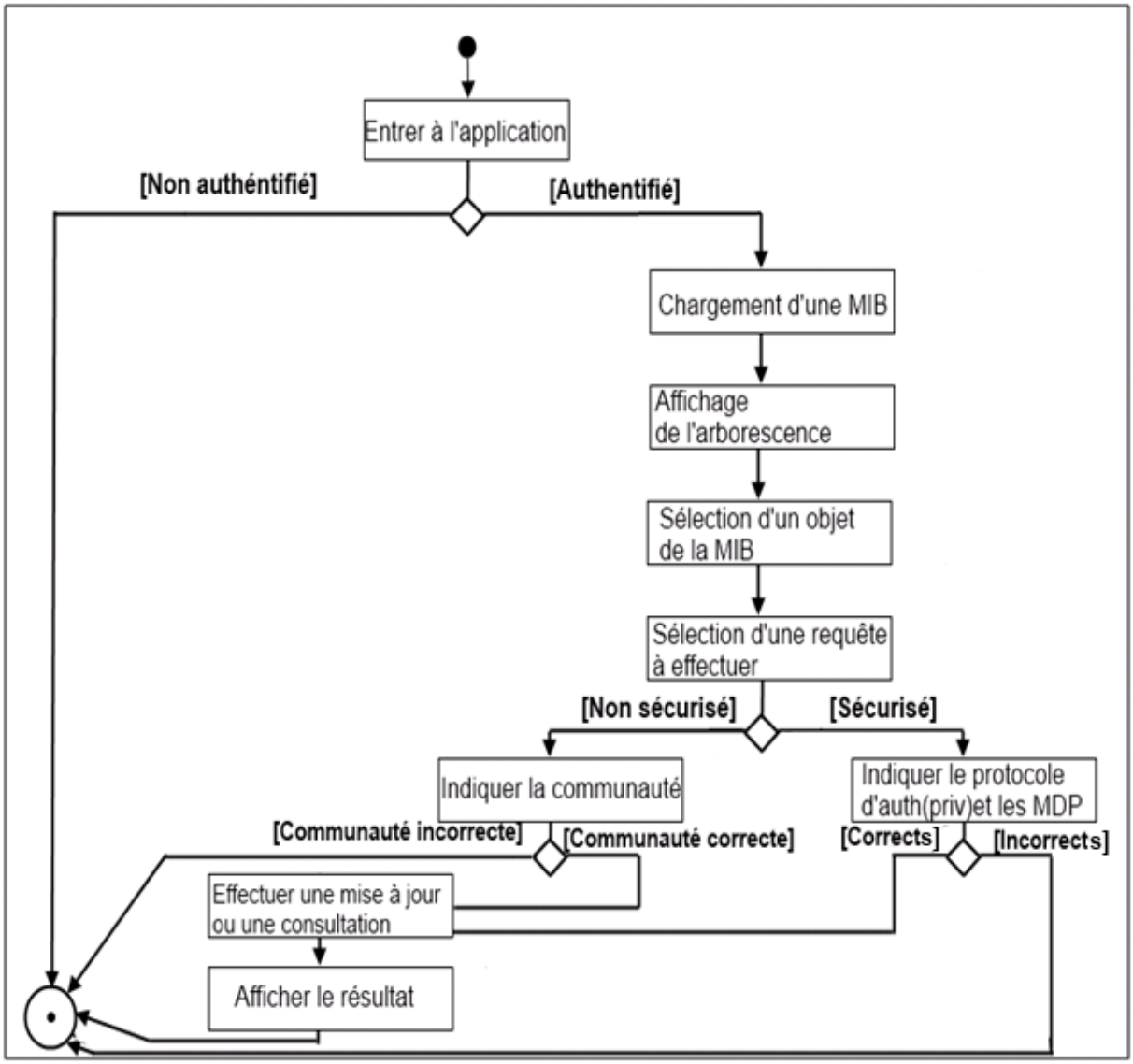

Figure 3.19 : Diagramme d'activité général

4. Phase2 : Conception de l'application

Afin de modéliser l'aspect statique de notre application, nous avons opté pour les diagrammes suivants :

- Diagrammes de paquetage.
- Diagramme de classe.

4.1. Diagramme de paquetage

Le paquetage est un mécanisme permettant de regrouper des classes, des interfaces et des packages.

La structuration d'un modèle en packages doit s'appuyer sur deux principes fondamentaux : *cohérence* et *indépendance.*

Le premier principe consiste à regrouper les classes qui sont proches d'un point de vue sémantique et le second principe s'efforce de minimiser les relations entre packages, c'est-à-dire plus concrètement les relations entre classes de packages différents. **[UCA]**

- **Relation d'import de package**

La relation d'import est monodirectionnelle, c'est-à-dire qu'elle comporte un package source et un package cible. Les classes du package source peuvent avoir accès aux classes du package cible.

La relation d'import entre deux paquetages s'exprime à l'aide d'une flèche en pointillés.

La figure suivante présente la relation d'import entre deux packages P1 et P2 contenant respectivement les classes A et B. (la classe A désire accéder à la classe B).

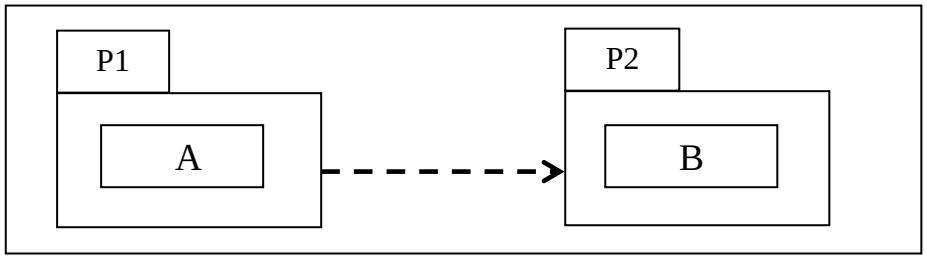

Figure 3.20 : Formalisme de représentation du diagramme de paquetage

Notre application comporte plusieurs packages.

« Requêtes SNMP » et « Gestion de la MIB » font appel au package « API SNMP » et « GUI », fait appel aux deux packages « Requêtes SNMP » et « Gestion de la MIB ».

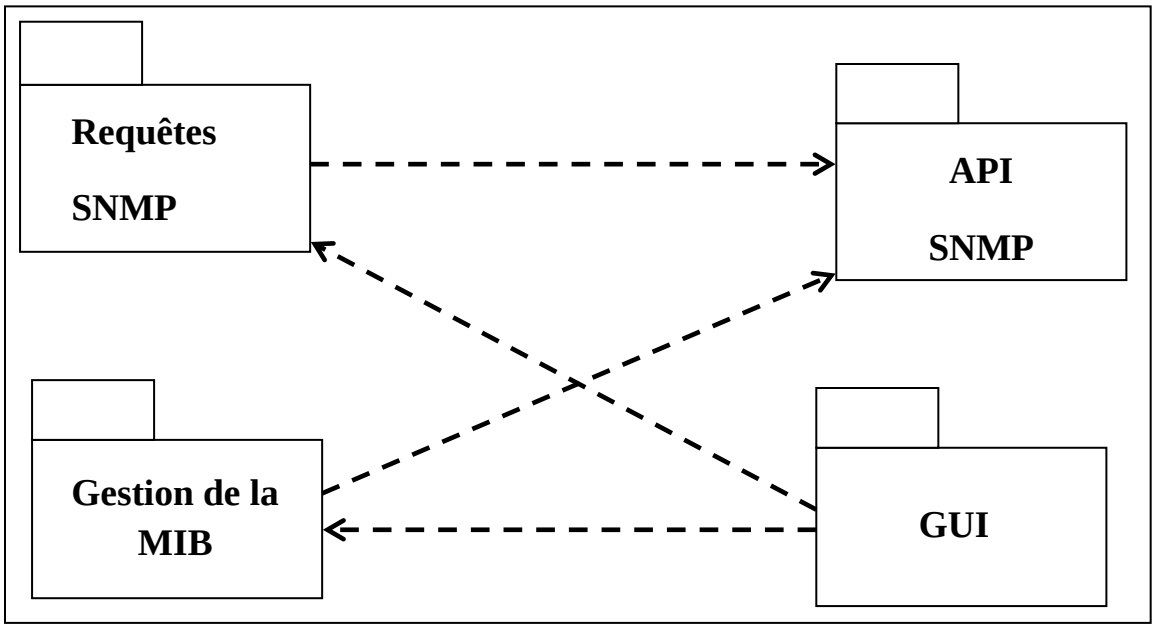

Figure 3.21 : Diagramme de paquetage de l'application

4.2. Diagramme de classes : permet de représenter des classes et leurs relations. Aussi, outre des classes, il peut contenir des types, des paquetages, des relations, voire des instances (objets et liens). Différents types de relations peuvent être exprimées dans un diagramme de classes, et notamment l'association (relation bidirectionnelle entre plusieurs classes).

Le diagramme de classes exprime la traçabilité directe entre un processus, ses travailleurs et ses entités (i.e. entre le cas d'utilisation et ses classes). **[UDV]**

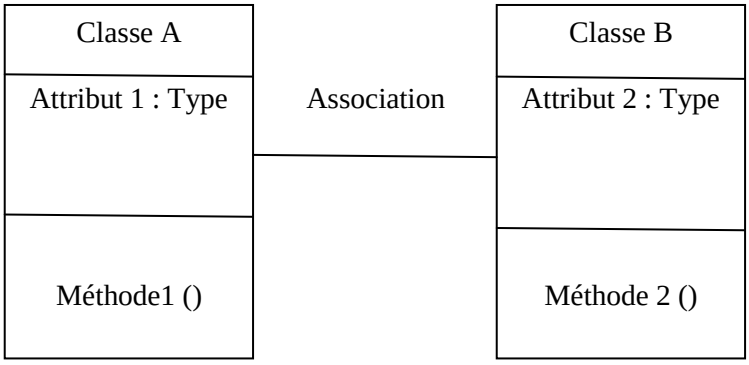

Figure 3.22 : Formalisme de représentation du diagramme de classes

Le diagramme de classe de l'application se base sur les règles de gestion qui suivent :

- Un utilisateur lance une ou plusieurs NMS (Station d'administration).
- Une NMS effectue une ou plusieurs requêtes SNMP.
- Une requête SNMP est exécutée par un ou plusieurs agents.
- Un agent envoie une notification à une ou plusieurs NMS.
- Une notification peut être de type Trap ou INFORM.
- Un agent maintient une ou plusieurs MIB.
- Une MIB est composée d'objets.
- Un objet peut être scalaire ou tabulaire.
- Une requête SNMP peut être de type : GET, GetNext, GetBulk ou SET.

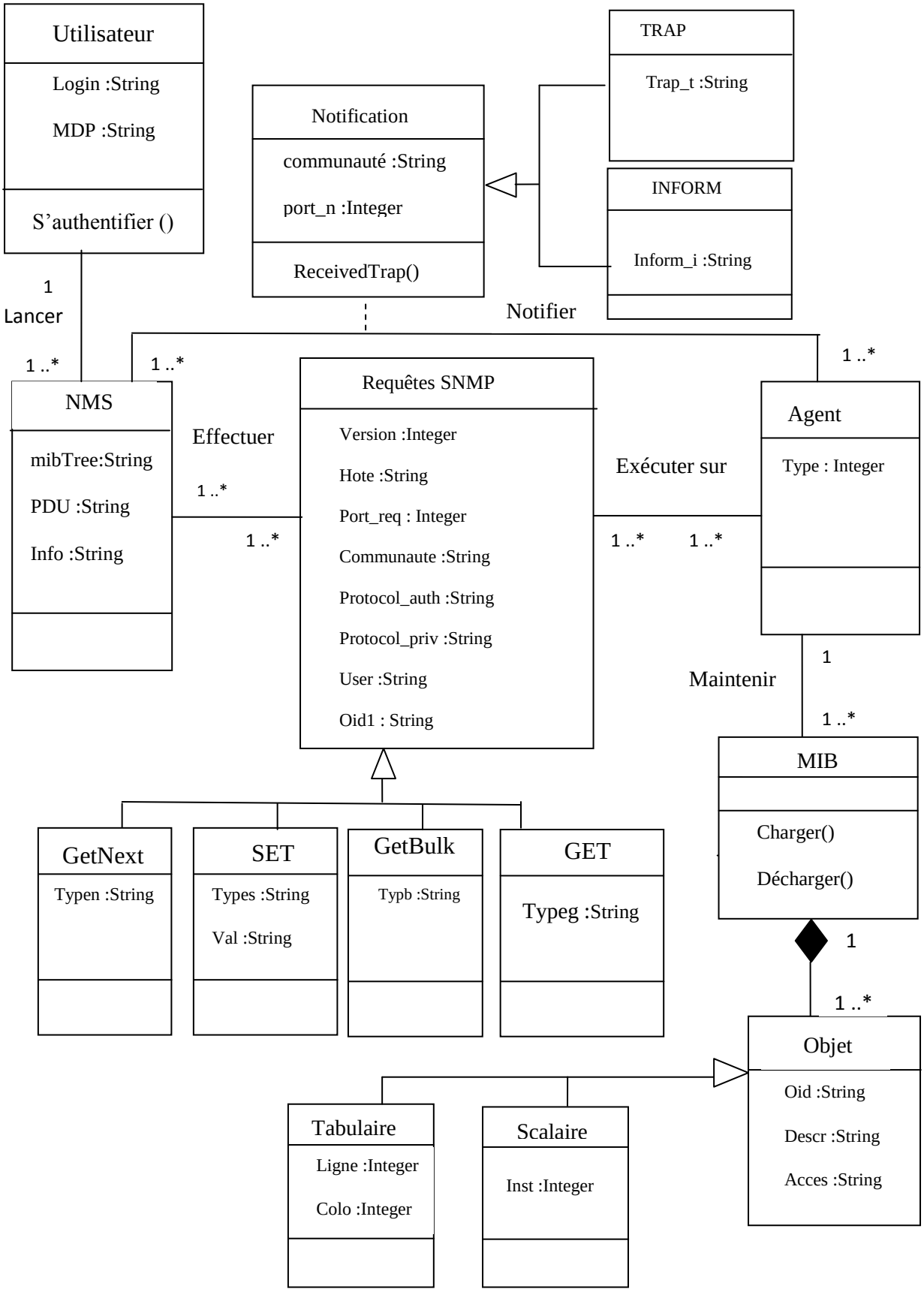

Figure 3.23 : Diagramme de classes

- **Dictionnaire de données**

Nom	Désignation
mibTree	Chemin de la MIB
PDU	Etat du PDU suite à une requête
Info	Information sur l'objet en cours
Version	Version de SNMP
hote	Destination de la requête
Port_req	Port d'écoute de requête
Port_n	Port d'écoute de notification
Protocol_auth	Protocole d'authentification
Protocol_priv	Protocole de privatisation
User	Utilisateur
Val	Valeur
Typer	Type de la requête
Trap_t	Indique qu'il s'agit d'un trap
Inform_i	Indique qu'il s'agit d'un inform
Typen	Dans le cas de GetNext
typeb	Dans le cas de GetBulk
typeg	Dans le cas de Get
types	Dans le cas de Set

Tableau 3.10 : Dictionnaire de données

5. Conclusion

Ce chapitre nous a permis de modéliser notre application en utilisant UML, et ainsi d'avoir une vue bien claire et précise sur l'application, ses comportements avec les diagrammes de cas d'utilisation, séquence et activité, ainsi que sa structure avec les diagrammes de paquetage et classes.

Le chapitre suivant sera consacré à la réalisation de l'application qui sera implémentée sous un langage orienté objet qui est le JAVA et où son fonctionnement sera expliqué.

Chapitre 4

Réalisation et tests

1. Introduction

Le chapitre précédent a été consacré à la modélisation de l'application, ce qui a facilité la tâche de l'implémentation. Nous avons opté pour l'Orienté Objet afin de mettre en œuvre cette dernière, vu qu'il est le plus approprié pour le développement d'applications similaires à la nôtre.

2. Architecture de l'application réalisée

Notre application se compose de plusieurs couches :

2.1. Couche présentation

Elle représente l'interface utilisateur de l'application (**GUI**). Ses principales fonctions sont :

- Activité de gestion réseau : Effectuer des requêtes de consultation ou mise à jour d'objets, en choisissant les protocoles de sécurité voulus, ainsi que l'écoute des traps et Inform en lançant l'écouteur.
- Permettre aux utilisateurs l'affichage des résultats selon plusieurs formes : texte, graphe ou tableaux.

2.2. Couche Gestion Réseau

Elle s'occupe de toutes les opérations de gestion réseau par SNMP. Elle utilise l'API AdventNet de SNMP.

- Lancement des requêtes SNMP : Module qui permet d'envoyer des requêtes SNMP à l'agent.
- Lancement ou arrêt du processus d'écoute de traps et d'inform à l'aide du gestionnaire de trap.
- Permettre à deux stations d'administration de communiquer par des notifications INFORM.

2.3. Couche Gestion de la MIB

C'est la couche qui a pour rôle la gestion de la MIB et prend donc en charge toutes les opérations qui peuvent être effectuées sur la MIB.

L'architecture de l'application est donnée dans la figure suivante :

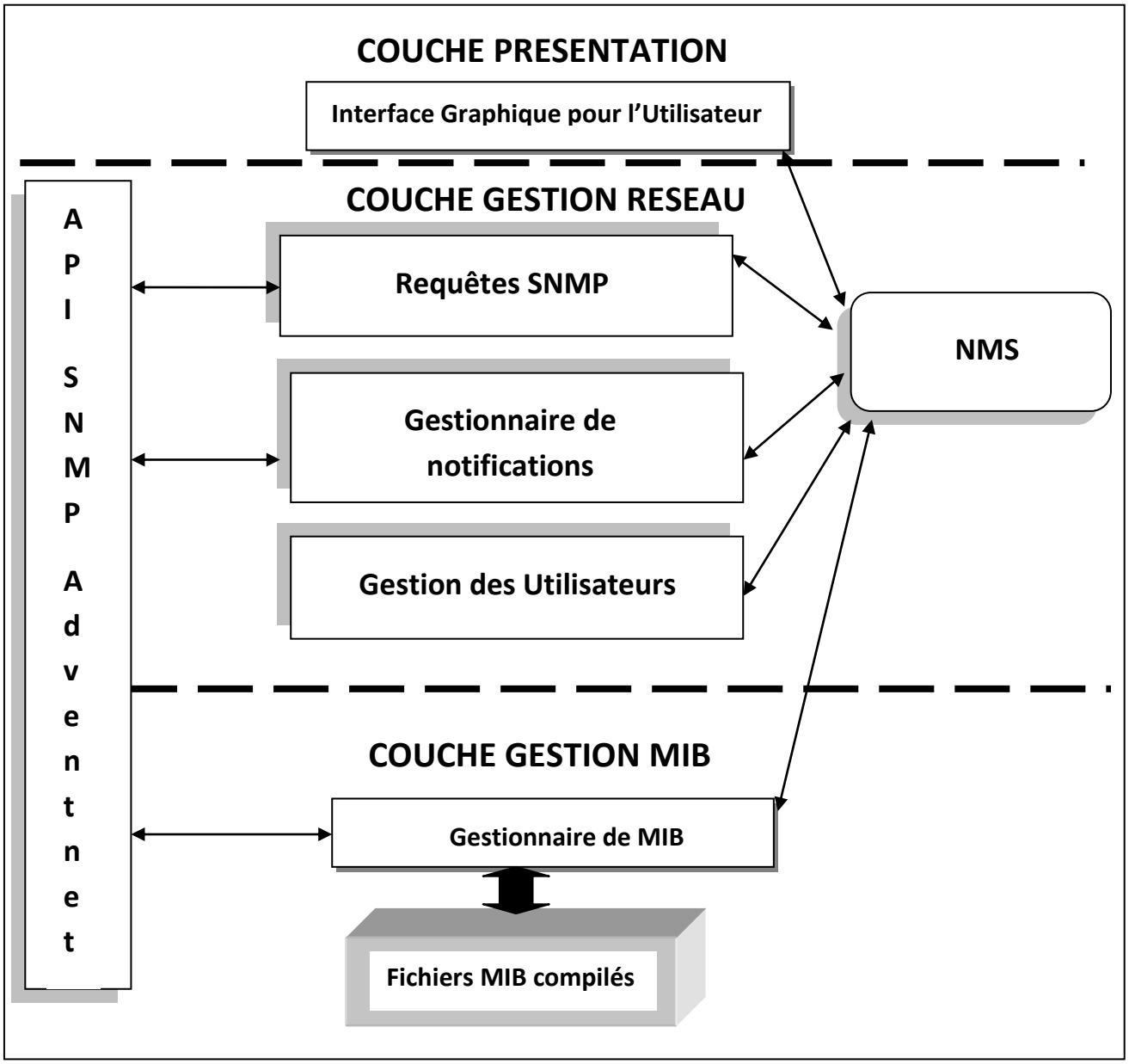

Figure 4.1 : Architecture générale de l'application

3. Diagramme de déploiement de l'application

3.1. Description du diagramme de déploiement

Il décrit l'architecture technique d'un système avec une vue centrée sur la répartition des composants dans la configuration d'exploitation. Cette architecture comprend des nœuds correspondant aux supports physiques (serveurs, routeurs…) ainsi que la répartition des artefacts logiciels (bibliothèques, exécutables…) sur ces nœuds. Les nœuds ainsi que la connexion qu'il existe entre eux constitue un réseau qui modélise cette architecture.

Un artefact est la spécification d'un élément physique qui est utilisé ou produit par le processus de développement du logiciel ou par le déploiement du système. C'est donc un élément concret comme par exemple : un fichier, un exécutable ou une table d'une base de données.

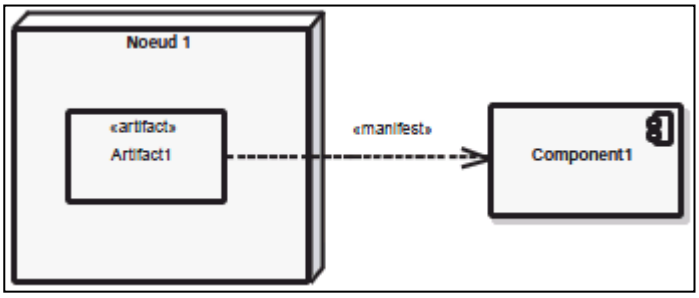

Figure 4.2 : Notation du diagramme de déploiement en UML2

3.2. Diagramme de déploiement de l'application

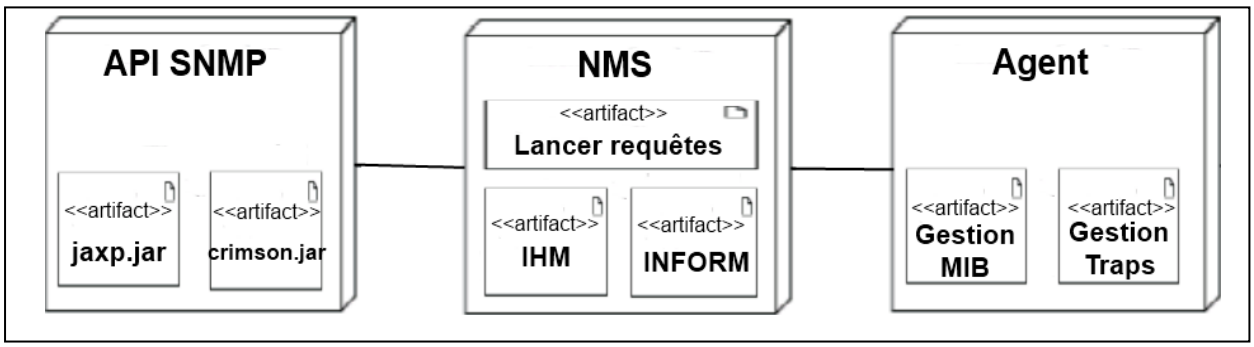

Figure 4.3 : Diagramme de déploiement de l'application

4. L'agent SNMPv3

L'application de gestion SNMP réalisée et appuyée sur la modélisation faite dans le chapitre précédent se base essentiellement sur l'installation préalable d'un agent SNMPv3 qui permet la création des utilisateurs, l'attribution de leurs droits d'accès ainsi que les communautés de lecture et d'écriture (S'il s'agit de SNMP non sécurisé).

- Afin d'effectuer une requête SNMPv1 ou SNMPv2, il est nécessaire de définir les communautés.

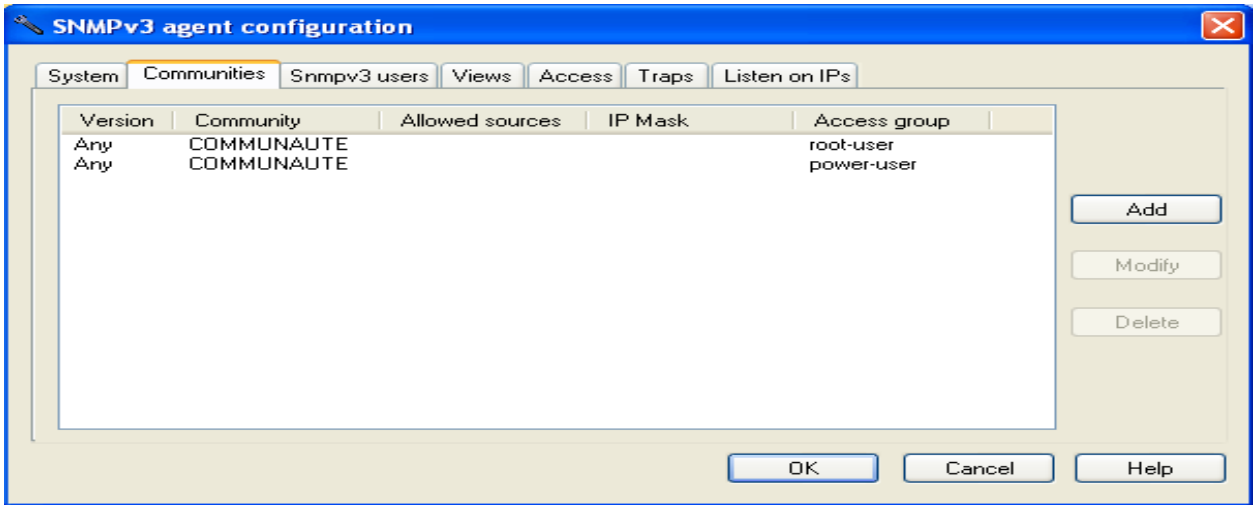

Figure 4.4 : Définition des communautés

- Par contre, pour effectuer des requêtes SNMPv3, la définition des utilisateurs est obligatoire.

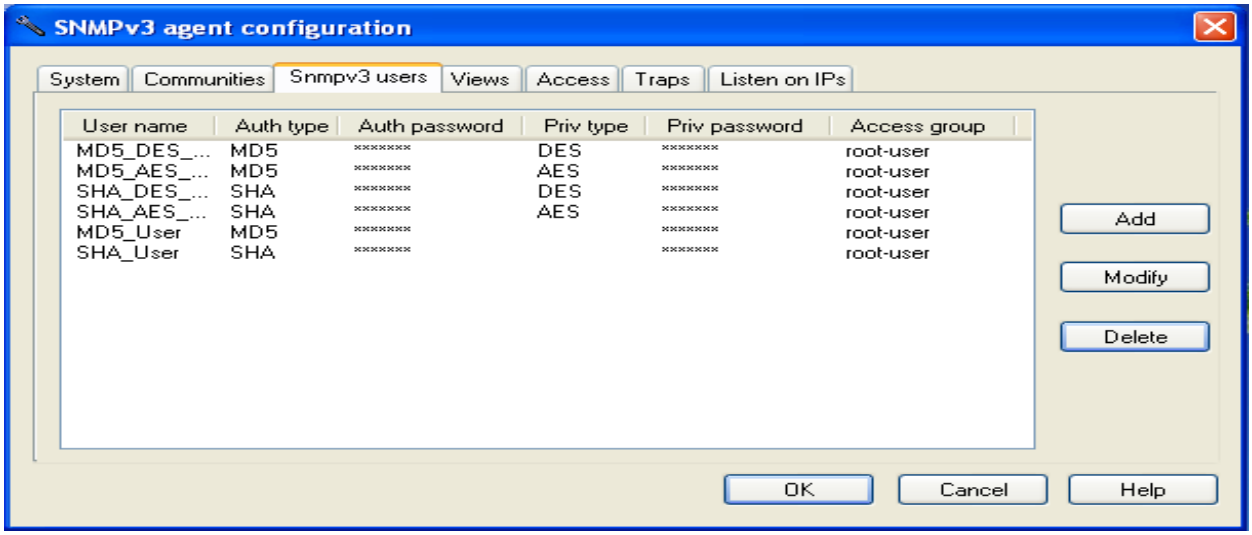

Figure 4.5 : Définition des utilisateurs SNMPv3

- **Exemple d'ajout d'un utilisateur**

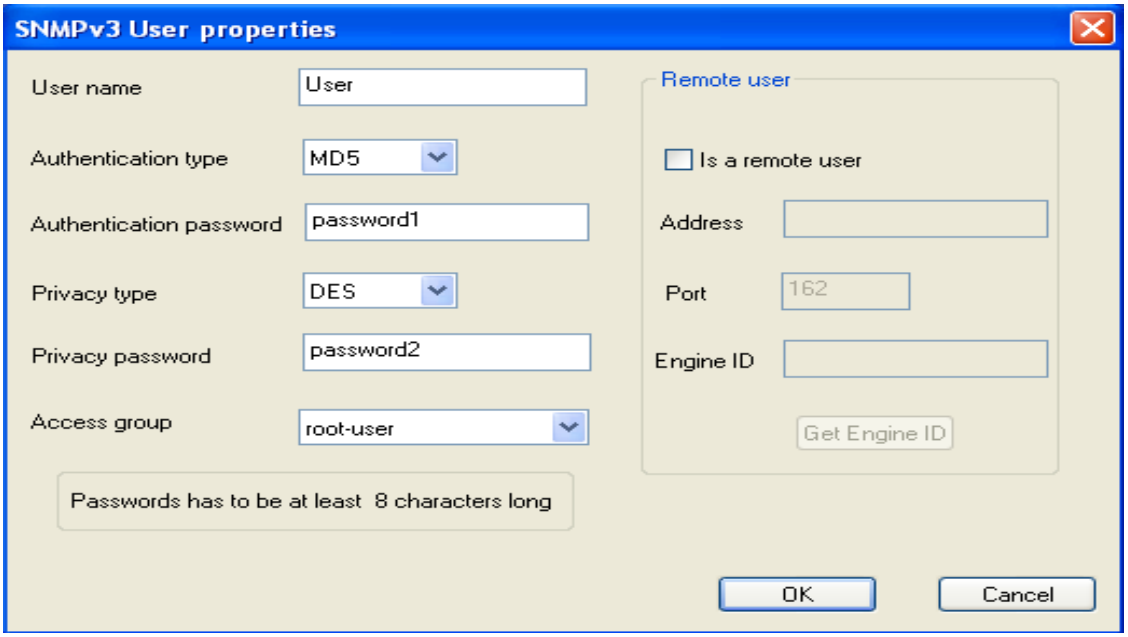

Figure 4.6 : Ajout d'un utilisateur

- En ce qui concerne la réception des traps, il est nécessaire de définir la destination de celle-ci, le port ainsi que la communauté.

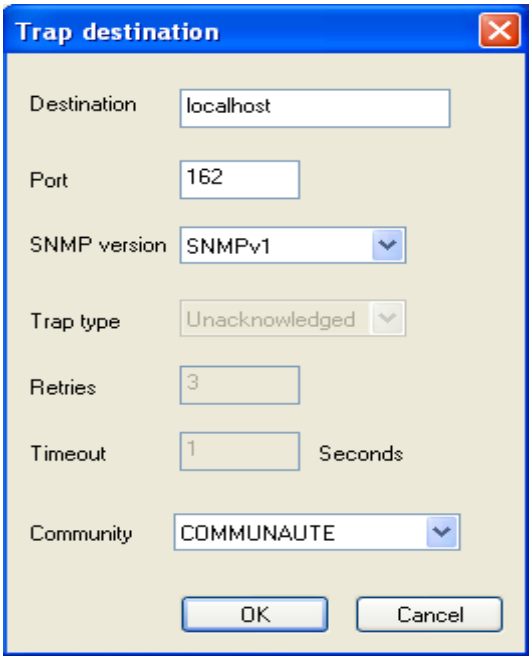

Figure 4.7 : Définition des traps

5. API utilisée pour la gestion SNMP

Plusieurs logiciels de gestion SNMP ont été mis en œuvre, on peut citer Net-SNMP qui est un ensemble d'outils permettant d'utiliser et de déployer le protocole SNMP (v1, v2c et v3…), ainsi que AdventNet qui représente un ensemble complet d'outils pour créer un environnement simulé.

L'API que nous avons employée pour la réalisation de notre application est SNMP API AdventNet.

L'API AdventNet SNMP est une API commercialisée, cependant, elle peut être utilisée pour usage académique, ou employée pour développer des applications de gestion de réseau.

Les développeurs d'applications de gestion de réseau peuvent employer la bibliothèque SNMP d'AdventNet pour construire des Applet, des composants, et des applications réparties d'EJB, de CORBA, et de RMI. La bibliothèque fournit les fonctions et les composants le plus généralement utilisés pour rendre le développement plus simple.

6. Présentation de l'application

Afin de développer notre application, nous nous sommes basés sur la modélisation faite dans le chapitre précédent.

Cette dernière a été réalisée en utilisant Eclipse de JAVA qui est un langage de programmation orienté objet.

6.1. Fonctions de l'application

Notre application permet de gérer un équipement sur lequel est installé un agent SNMP, l'accès à cet équipement se fait par son adresse IP et par défaut et comme mentionné dans les chapitres précédents, avec le port 161.

Les principales fonctionnalités de l'application de gestion sont :

- Chargement et déchargement d'une MIB.
- Enregistrement des travaux réalisés.
- Choix de version à utiliser pour la gestion (v1/v2 ou v3).

- Ecoute des traps émis par l'agent.
- Envoi de notifications INFORM à une autre station de gestion.
- Affichage de la table d'un objet tabulaire.
- Affichage de la description des objets de façon détaillée.
- Affichage l'état du PDU suite à une requête émise.
- Possibilité d'effectuer des opérations de consultation telles que : GET, GetNext et GetBulk ; ou de mise à jour comme SET.
- Afficher le graphe d'évolution d'état d'une variable de type entier (Polling).

6.2. Présentation de l'interface

- **Présentation de la palette**

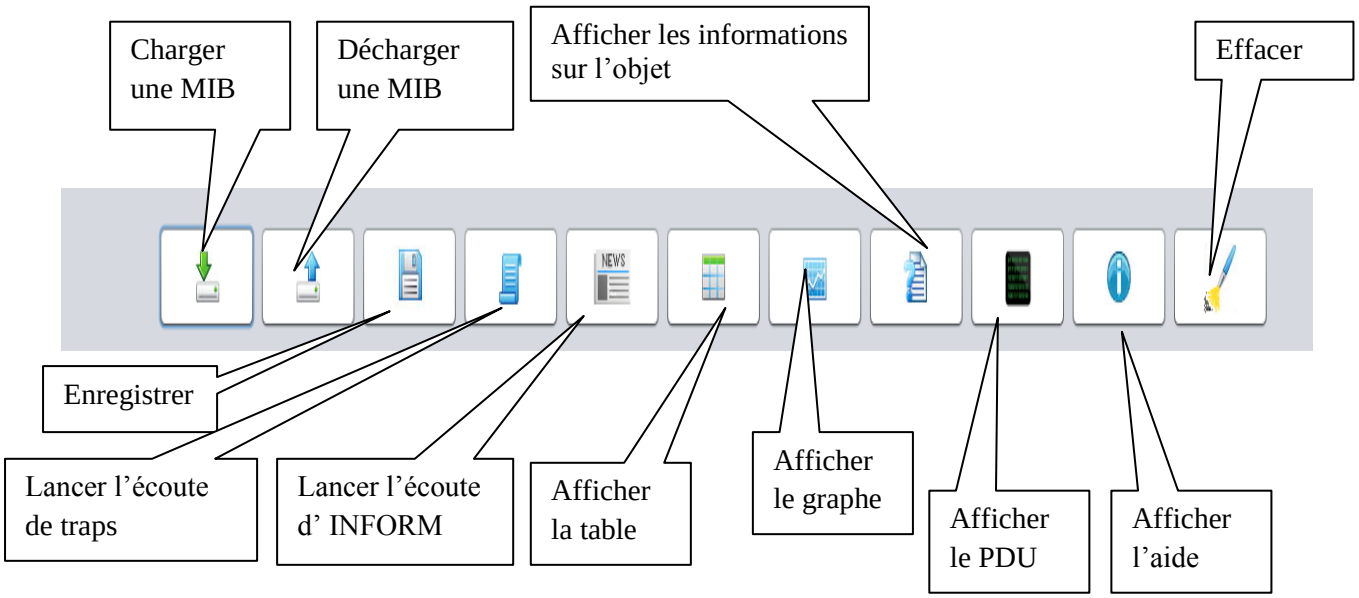

Figure 4.8 : Palette de l'application

- **Requête GET sécurisée avec authentification et privatisation sur les variables SysDescr et SysUpTime**

L'exemple illustre une requête GET SNMPv3 sur les variables SysUpTime et SySDescr avec authentification en utilisant le protocole MD5 et privatisation avec le protocole AES.

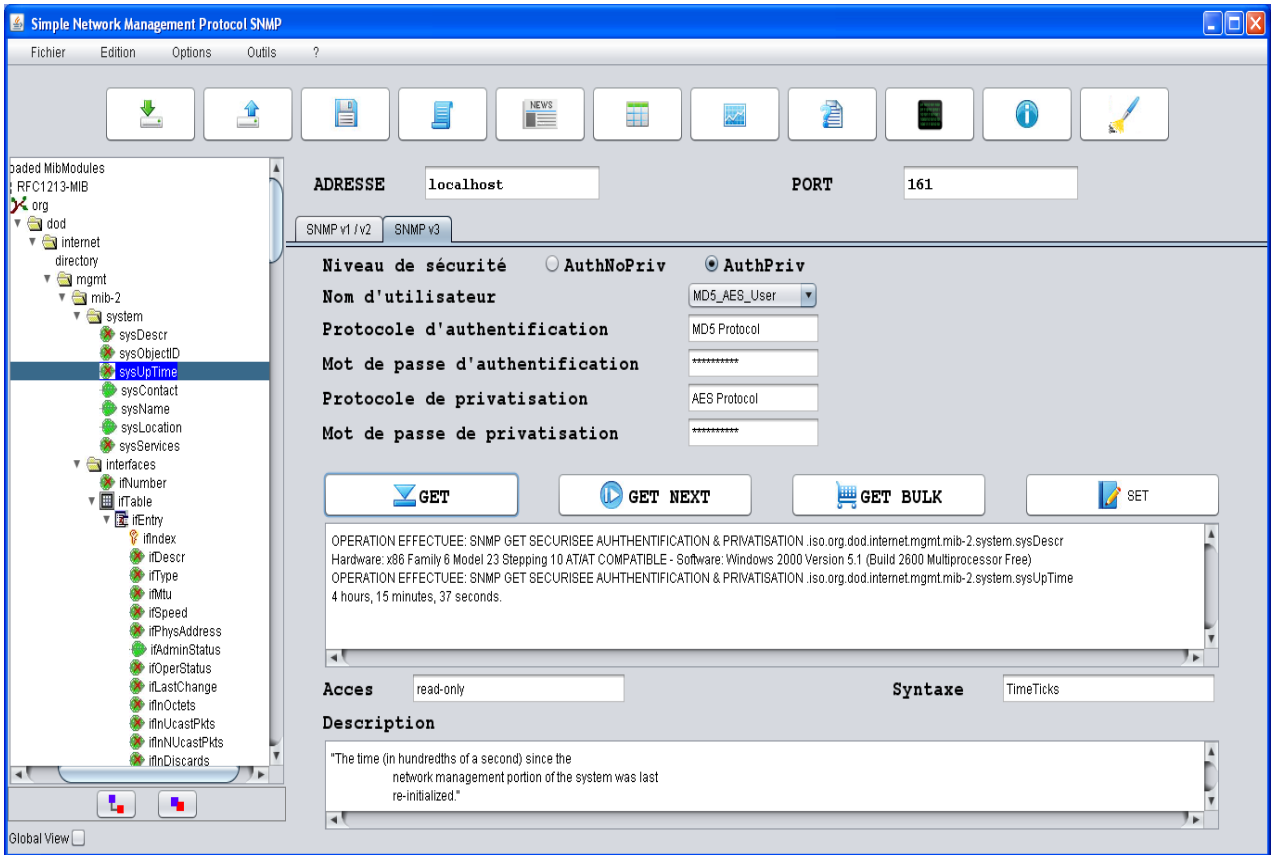

Figure 4.9 : Requête GET sécurisée

- **Requête SET sécurisée avec authentification sur la variable SysContact**

L'exemple illustre une requête Set SNMPv3 sur la variable SysContact avec authentification en utilisant le protocole SHA.

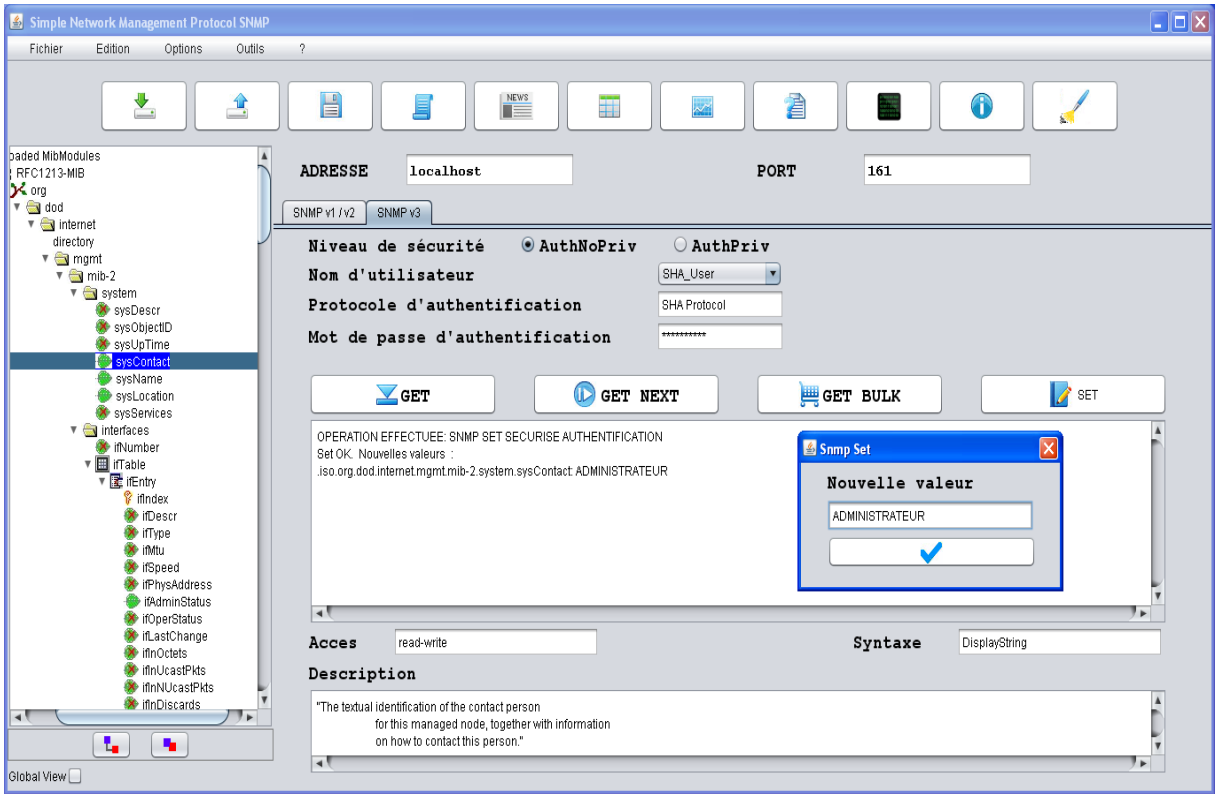

Figure 4.10 : Requête SET sécurisée

- **Lancement de l'écouteur de traps**

Cette fenêtre illustre l'activation de l'écouteur de traps, la liste des traps apparait suite à un événement notifié par l'agent.

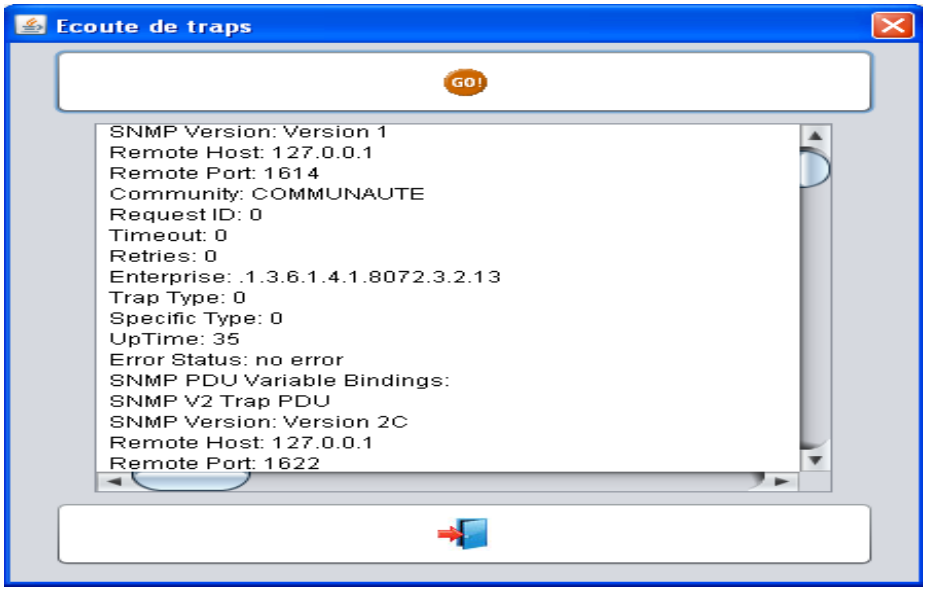

Figure 4.11 : Liste des traps reçus

- **Dessin du graphe de variation de la variable SysUpTime**

Le polling est lancé sur une variable de type entière tel que SysUpTime. Le résultat est illustré dans le graphe suivant :

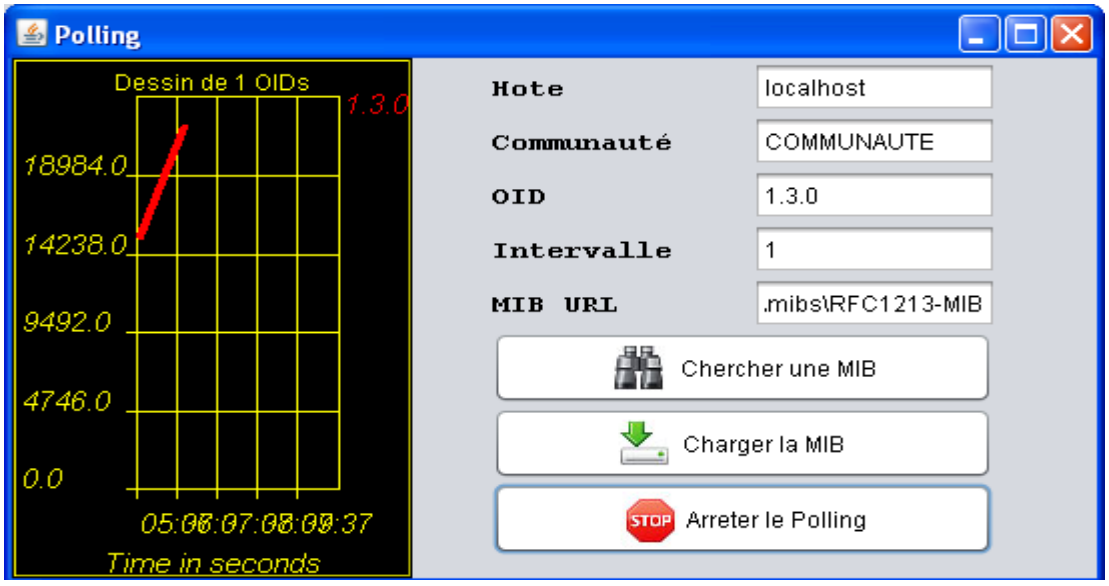

Figure 4.12 : Graphe de variation

7. Conclusion

Notre application a porté sur la réalisation d'un outil permettant la gestion d'équipements réseaux basée sur la sécurité apportée par le protocole SNMPv3, offrant la possibilité de consulter ou mettre à jour des objets appartenant à des MIB maintenues par des agents installés sur les équipements gérés.

Conclusion et perspectives

Les réseaux informatiques touchent de plus en plus notre vie courante et les services offerts par ceux-ci sont devenus indispensables. Pour assurer que les services rendus soient convenables, il est nécessaire de surveiller le réseau et d'agir quand une erreur se produit.

SNMP est le standard incontournable dans le domaine de l'administration de réseaux, il permet à la plate-forme de gestion de récupérer des informations sur les éléments du réseau et aussi de changer les paramètres sur ce dernier, en plus d'offrir la possibilité de gérer des alarmes sur des événements imprévus.

Le projet a consisté en l'élaboration et mise en œuvre d'une application de gestion réseau basée sur le protocole SNMP, incluant la sécurité des échanges entres les stations d'administration et les agents (SNMPv3), afin de permettre à la conversation d'être à l'abri d'intentions malveillantes des espions du réseau.

Ce travail a eu énormément d'apport sur nos connaissances, notamment en découvrant les différentes techniques de gestion réseau, et en élaborant une étude approfondie sur le protocole de gestion SNMP. De plus, il nous a permis de mettre en pratique nos connaissances en langage de modélisation tel que UML d'une part, et la programmation orientée objet de l'autre part.

Comme perspectives de ce travail, noue envisageons de développer des agents SNMP supportant toutes les fonctionnalités de la troisième version et compatibles avec tous les équipements réseau existants.

Annexe

1. IETF

L'Internet Engineering Task Force, abrégée IETF, est un groupe informel, international, ouvert à tout individu, qui participe à l'élaboration de standards Internet. L'IETF produit la plupart des nouveaux standards d'Internet.

2. RFC

Les RFCs (*Request For Comments*) sont un ensemble de documents qui font référence auprès de la Communauté Internet et qui décrivent, spécifient, aident à l'implémentation, standardisent et débattent de la majorité des normes, standards, technologies et protocoles liés à Internet et aux réseaux en général.

- Quelques RFC utilisées

RFC1441 : Introduction to version 2 of the Internet-standard Network Management Framework

RFC 1445 : Administrative Model for version 2 of the Simple Network Management Protocol (SNMPv2)

RFC 1449 : Transport Mappings for version 2 of the Simple Network Management Protocol (SNMPv2)

RFC 768 : User Datagram Protocol

RFC 3826 : The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User based Security Model.

3. Le protocole UDP

User Datagram Protocol (UDP) est un des principaux protocoles de télécommunication utilisés par Internet. Il fait partie de la couche transport de la pile de protocole TCP/IP : dans l'adaptation approximative de cette dernière au modèle OSI, il appartiendrait à la couche 4, comme TCP. Il est détaillé dans la RFC 768.

Le rôle de ce protocole est de permettre la transmission de données et de manière très simple entre deux entités, chacune étant définie par une adresse IP et un numéro de port. Contrairement au protocole TCP, il fonctionne sans négociation : il n'existe pas de procédure de connexion préalable à l'envoi des données (le handshaking). Donc UDP ne garantit pas la bonne livraison des datagrammes à destination, ni de leur ordre d'arrivée, il est également possible que des datagrammes soient reçus en plusieurs exemplaires.

L'intégrité des données est assurée par une somme de contrôle sur l'en-tête. L'utilisation de cette somme est cependant facultative en IPv4 mais obligatoire avec IPv6. Si un hôte n'a pas calculé la somme de contrôle d'un datagramme émis, la valeur de celle-ci est fixée à zéro. La somme de contrôle inclut les adresses IP source et destination.

La nature d' UDP le rend utile pour transmettre rapidement de petites quantités de données, depuis un serveur vers de nombreux clients ou bien dans des cas où la perte d'un datagramme est moins gênante que l'attente de sa retransmission.

4. SHA-1

SHA-1 (Secure Hash Algorithm) est une fonction de hachage cryptographique conçue par la National Security Agency des États-Unis (NSA), et publiée par le gouvernement des États-Unis comme un standard fédéral de traitement de l'information (Federal Information Processing Standard du National Institute of Standards and Technology (NIST)). Elle produit un résultat (appelé « hash » ou condensat) de 160 bits.

Le SHA-1 prend un message d'un maximum de 2^64 bits en entrée. Son fonctionnement est similaire à celui du MD4 ou MD5 de Ronald Rivest. Quatre fonctions booléennes sont définies, elles prennent 3 mots de 32 bits en entrée et calculent un mot de 32 bits. Une fonction spécifique de rotation est également disponible, elle permet de déplacer les bits vers la gauche (le mouvement est circulaire et les bits reviennent à droite). Une de ces rotations n'était pas présente dans le SHA-0, elle permet de casser certaines caractéristiques linéaires dans la structure. Cela permet d'éviter une attaque sur les bits neutres décrite par Eli Biham, technique reprise pour calculer la collision complète sur SHA-0.

Le SHA-1 commence par ajouter à la fin du message un bit à 1 suivi d'une série de bits à 0, puis la longueur du message initial (en bits) codée sur 64 bits. La série de 0 a une longueur telle que la

séquence ainsi prolongée a une longueur multiple de 512 bits. L'algorithme travaille ensuite successivement sur des blocs de 512 bits.

Pour chaque bloc, l'algorithme calcule 80 tours successifs et applique une série de transformations sur l'entrée. La première étape consiste à calculer 80 valeurs sur 32 bits. Les 16 premières valeurs sont obtenues directement à partir du bloc « message » en entrée. Les 64 autres sont calculées successivement. Le SHA-1 les obtient grâce à une rotation qui est appliquée sur le résultat d'un XOR, il utilise pour cela 4 mots obtenus dans les itérations précédentes. On définit ensuite 5 variables qui sont initialisées avec des constantes (spécifiées par le standard), le SHA-1 utilise encore 4 autres constantes dans ses calculs. Si un bloc de 512 bits a déjà été calculé auparavant, les variables sont initialisées avec les valeurs obtenues à la fin du calcul sur le bloc précédent.

Il s'ensuit 80 tours qui alternent des rotations, des additions entre les variables et les constantes. Selon le numéro du tour, le SHA-1 utilise une des quatre fonctions booléennes. L'une de ces fonctions est appliquée sur 3 des 5 variables disponibles. Les variables sont mises à jour pour le tour suivant grâce à des permutations et une rotation. En résumé, le SHA-1 change sa méthode de calcul tous les 20 tours et utilise les sorties des tours précédents.

À la fin des 80 tours, on additionne le résultat avec le vecteur initial. Lorsque tous les blocs ont été traités, les cinq variables concaténées ($5 \times 32 = 160$ bits) représentent la signature.

5. MD5

L'algorithme MD5, pour Message Digest 5, est une fonction de hachage cryptographique qui permet d'obtenir l'empreinte numérique d'un fichier. Il a été inventé par Ronald Rivest en 1991.

6. DES

Le Data Encryption Standard (DES) est un algorithme de chiffrement symétrique (chiffrement par bloc) utilisant des clés de 56 bits. Son emploi n'est plus recommandé aujourd'hui, du fait de sa lenteur à l'exécution et de son espace de clés trop petit permettant une attaque systématique en un temps raisonnable. Quand il est encore utilisé c'est généralement en Triple DES, ce qui ne fait rien pour améliorer ses performances. DES a notamment été utilisé dans le système de mots de passe UNIX.

Le premier standard DES est publié par FIPS le 15 janvier 1977 sous le nom FIPS PUB 46. La dernière version avant l'obsolescence date du 25 octobre 1999 FIPS PUB 46-3.

L'algorithme DES transforme un bloc de 64 bits en un autre bloc de 64 bits. Il manipule des clés individuelles de 56 bits, représentées par 64 bits (avec un bit de chaque octet servant pour le contrôle de parité). Ce système de chiffrement symétrique fait partie de la famille des chiffrements itératifs par blocs, plus particulièrement il s'agit d'un schéma de Feistel (du nom de Horst Feistel à l'origine du chiffrement Lucifer).

D'une manière générale, on peut dire que DES fonctionne en trois étapes :

- permutation initiale et fixe d'un bloc (sans aucune incidence sur le niveau de sécurité).
- le résultat est soumis à 16 itérations d'une transformation, ces itérations dépendent à chaque ronde d'une autre clé partielle de 48 bits. Cette clé de ronde intermédiaire est calculée à partir de la clé initiale de l'utilisateur (grâce à un réseau de tables de substitution et d'opérateurs XOR). Lors de chaque ronde, le bloc de 64 bits est découpé en deux blocs de 32 bits, et ces blocs sont échangés l'un avec l'autre selon un schéma de Feistel. Le bloc de 32 bits ayant le poids le plus fort (celui qui s'étend du bit 32 au bit 64) subira une transformation.
- le dernier résultat de la dernière ronde est transformé par la fonction inverse de la permutation initiale.

DES utilise huit tables de substitution (les S-Boxes) qui furent l'objet de nombreuses controverses quant à leur contenu. On soupçonnait une faiblesse volontairement insérée par les concepteurs. Ces rumeurs furent dissipées au début des années 1990 par la découverte de la cryptanalyse différentielle qui démontra que les tables étaient bien conçues.

7. AES

Advanced Encryption Standard ou AES (standard de chiffrement avancé), aussi connu sous le nom de Rijndael, est un algorithme de chiffrement symétrique. Il remporta en octobre 2000 le concours AES, lancé en 1997 par le NIST et devint le nouveau standard de chiffrement pour les organisations du gouvernement des États-Unis. Il a été également approuvé par la NSA (National Security Agency) pour les informations top secrètes.

L'algorithme prend en entrée un bloc de 128 bits (16 octets), la clé fait 128, 192 ou 256 bits. Les 16 octets en entrée sont permutés selon une table définie au préalable. Ces octets sont ensuite placés dans une matrice de 4x4 éléments et ses lignes subissent une rotation vers la droite.

L'incrément pour la rotation varie selon le numéro de la ligne. Une transformation linéaire est ensuite appliquée sur la matrice, elle consiste en la multiplication binaire de chaque élément de la matrice avec des polynômes issus d'une matrice auxiliaire, cette multiplication est soumise à des règles spéciales selon GF(2^8) (groupe de Galois ou corps fini). La transformation linéaire garantit une meilleure diffusion (propagation des bits dans la structure) sur plusieurs tours.

Finalement, un XOR entre la matrice et une autre matrice permet d'obtenir une matrice intermédiaire. Ces différentes opérations sont répétées plusieurs fois et définissent un « tour ». Pour une clé de 128, 192 ou 256, AES nécessite respectivement 10, 12 ou 14 tours.

8. Replay attack

Comme son nom l'indique, le principe de la "replay attack" consiste à rejouer un paquet. L'assaillant intercepte un paquet valide sur le réseau. Il le renvoie au moment opportun afin d'obtenir des informations de la part du récepteur original du paquet.

9. MIB-2

La RFC 1213 décrit le module Mib-2 qui recense un certain nombre d'objets de base devant être supportés par tout agent SNMP, tels que: Nom de l'administrateur, localisation, liste des interfaces réseau, vitesse, octets transmis…etc.

Groupe	**OID**	**Description**
System	{ mib-2 1 }	Système géré : son nom, le type d'équipement, etc.
Interface	{ mib-2 2}	Interfaces réseau. La table ifTable contient la description, l'état et les statistiques des interfaces.
at	{ mib-2 3 }	Address Translation : atTable contient la table des adresses MAC (Media Access Control) et réseau. Ce groupe est peu utilisé (*deprecated*) car remplacé par les équivalents spécifiques à un protocole réseau.
IP	{ mib-2 4 }	IP : configuration générale, statistiques, table d'adresses

		(ipTable), de routage (ipRouteTable), table ARP (AddressResolutionProtocol) (ipNet-ToMediaTable). ipRouteTable (21) est en principe remplacée par ipForward (24) (défini dans la RFC 1354) car indexée par l'adresse destination et donc ne supportant pas les routes multiples.
ICMP	{ mib-2 5 }	ICMP (Internet Control Message Protocol) : statistiques.
TCP	{ mib-2 6 }	TCP : configuration, statistiques générales et table des connexions (tcp-ConnTable).
UDP	{ mib-2 7 }	UDP : statistiques et table des *listeners* (udpTable).
EGP	{ mib-2 8 }	EGP (Exterior Gateway Protocol) : statistiques et table des voisins (egpNeighTable).
CMOT	{ mib-2 9 }	CMIP over TCP/IP : obsolète. Seul l'OID est réservé dans mib-2.
Transmission	{ mib-2 10 }	Ce groupe est prévu pour « raccrocher » d'autres modules de MIB qui concernent des médias de transmission plus spécifiques qui viennent compléter les informations contenues dans le groupe interface. Par exempte : dot3 (7) décrit le média Ethernet, tandis que dot5 (9) décrit Token-ring.
SNMP	{ mib-2 11 }	SNMP : statistiques.

Références

Bibliographie

[BCL] Bernard Cousin, *Administration du réseau*, Université Rennes I, 1998.

[EYR] Pascal Roques, *UML 2 par la pratique*, EYROLLES, 5ème édition, 2006.

[FAT] FARAH Zoubeyr, TOULOUM Karim : Conception et Réalisation d'un Outil d'Administration Réseaux TCP/IP basé sur le Protocole SNMP, Mémoire d'Ingéniorat d'Etat en Informatique, Université Abderrahmane Mira Bejaia, 2005.

[FFO] http://kindman.amc-os.com/pub/rapport_fiifo4_snmp.pdf, 2012.

[MAI] Ylian St-Hilaire, SNMPV3-Modulaire : Une méthode de conception et de mise en œuvre d'un protocole de gestion de réseau, Mémoire en Maîtrise Informatique, Université du Québec à Montréal, 1998.

[MLA] N.Gaertner et P. Muller, Modélisation objet avec UML, EYROLLES 2004.

[NAG] Thiery Briche, Mathieu Voland, *Les outils d'administration et de supervision réseau : Nagios*, 2004.

[NES] http://uuu.enseirb.fr/~kadionik/embedded/snmp/annexe.html , 2012.

[PRE] Michel Gardie, *La couche présentation La syntaxe ASN.1*, 2004.

[SDR] http://www.frameip.com/snmp/ , 2012.

[SVH] Yves Bertsch, Frederic St Marcel, « Administration réseau - système SNMPv1, SNMPv2, SNMPv3 et HTTP » - *Technologies réseaux avancées*-, 2001.

[UDV] Xavier Blanc, Isabelle Mounier, *UML 2 pour les développeurs*, EYROLLES, 2006.

[UCA] Joseph Gabay, David Gabay, *UML 2 Analyse et Conception*, Paris, DUNOD, 2008.

[OLW] Olivier Willm « Administration de réseaux informatiques : Le protocole SNMP » -*Techniques de l'ingénieur*- H2840, 2003.

[WLY] Jean Willy OLENGA, Implémentation d'une application de gestion de réseau basée sur le protocole SNMP, Université de Kinshasa, 2007. **s**

Résumé

Le projet consiste en la réalisation d'un outil de gestion réseau basé sur le protocole SNMP qui est considéré comme étant le standard le plus utilisé en matière de gestion.

SNMP (Simple Network Management Protocol) est un protocole d'administration réseau qui permet à un poste de contrôle d'obtenir et de modifier divers éléments de configuration d'équipements réseau.

La 3ème version de SNMP a été choisie pour ses apports en sécurité : Authentification et Cryptage en plus des fonctions de base offertes par ce protocole.

L'application développée permet de gérer les équipements réseau avec SNMP incluant la mise à jour et consultation des variables des équipements gérés.

La modélisation de l'application de gestion a été faite en utilisant le langage UML, puis implémentée sous le langage JAVA dans l'environnement de développement Eclipse.

Mots-clés : SNMP, UML, JAVA, Eclipse.

Abstract

The purpose of this project is the realization of a network management tool based on SNMP; which is known as the most used protocol on network administration.

SNMP is a network management protocol which provides services of getting and changing several elements' statements of the managed network devices.

The 3rd version of SNMP is chosen for its security contributions: Authentication and encryption, added to the basic functions offered by this protocol.

The application permit to manage network equipments using SNMP, including consultation and updates on managed equipments' variables.

The modeling of the management application is done using the UML language, then implemented under JAVA language in the Eclipse development environment.

Key words: SNMP, UML, JAVA, Eclipse.