Contrôle des réseaux ad hoc à base de politiques

Yacine M. GHAMRI-DOUDANE, Sidi-Mohammed SENOUCI et Guy PUJOLLE

Laboratoire LIP6 - Université Pierre et Marie CURIE
08, Rue du capitaine Scott, 75015 Paris, FRANCE
Tél. : 01.44.27.61.88, Fax : 01.44.27.74.95
{ghamri, senouci, pujolle}@rp.lip6.fr

Résumé

Les réseaux ad hoc connaissent, actuellement, de plus en plus de succès dû à la facilité de leur déploiement et aux économies qu'ils permettent d'effectuer. Malgré ces avantages, les réseaux ad hoc présentent quelques failles liées à la faiblesse de la sécurisation des communications ainsi qu'au manque de garantie de qualité de service (QoS). La tendance actuelle pour résoudre ce genre de problèmes dans les réseaux fixes, est réalisée par l'intermédiaire d'un modèle par politiques. Ce modèle repose sur l'utilisation d'un serveur de politique pour mettre en place, gérer et configurer les différents services offerts par le réseau. Nous proposons dans ce travail d'adapter ce modèle par politique pour les réseaux ad hoc afin d'outre passer ces problèmes de sécurité et de QoS. Cette adaptation est nécessaire vu que ce genre de réseau ne repose sur aucune infrastructure et n'est composé que d'équipements mobiles. La solution proposée est une solution robuste traitant tous les cas de figures qui peuvent survenir dans un réseau ad hoc. Des extensions du protocole COPS-SLS ont également été proposées. Nos propositions sont compatibles aux travaux menés dans le cadre du contrôle des réseaux à base de politiques.

Mots clés : *réseaux ad hoc, contrôle par politiques, COPS.*

1. Introduction

Les progrès réalisés dans le domaine des réseaux locaux sans fils WLAN (Wireless LAN) ont ouvert de nouveaux horizons dans le domaine des télécommunications surtout depuis l'apparition de produits commerciaux basés sur la norme IEEE 802.11 [1]. Les réseaux ad hoc se distinguent des autres formes de réseaux sans fil par une absence totale d'infrastructure fixe et d'administration centralisée et ne sont constitués que de nœuds libres de se déplacer à leur guise. Ces réseaux connaissent, actuellement, de plus en plus de succès dû à la facilité de leur déploiement et aux économies qu'ils permettent d'effectuer. Un certain nombre d'industriels (Ericsson, IBM, Intel...) et d'organisations (IEEE, ITU, IETF) planchent sur le sujet afin d'établir de nouveaux standards qui permettent de supporter ce genre de réseaux. Les débits atteints aujourd'hui par ces réseaux permettent d'exécuter des applications complexes, telles que les applications multimédia (vidéo conférence, visiophonie,…), nécessitant des garanties sur le débit, le délai ou encore la gigue. Or ces applications consomment énormément de ressources et ne permettent pas une utilisation efficace et équitable du support de communication surtout quand elles coexistent avec des services de données caractérisés par des transferts en rafales (Burst). Les travaux menés autour de Mobile IP ou de l'UMTS ne peuvent être transposés directement aux réseaux ad hoc. Ces réseaux ne disposent pas encore de protocoles de qualité de service standards adaptés à leurs spécificités. Dans [2] l'auteur présente un état de l'art des recherches dans ce domaine.

D'un autre coté, Les standards de ces réseaux mobiles ad hoc fournissent actuellement une sécurité de base optionnelle. En effet, en plus du fait qu'elle soit optionnelle, la sécurité n'est assurée qu'entre les stations et non de bout en bout. De plus les protocoles qui sont utilisés pour assurer cette sécurité, comme WEP (Wired Equivalent Privacy) pour 802.11, présentent quelques failles (Par exemple la clé de chiffrage WEP peut être facilement déduite à cause des faiblesses de l'algorithme RC4) [3].

La nécessité de fournir un mécanisme capable de partager équitablement les ressources, de garantir une qualité de service (QoS) et de fournir une sécurité d'utilisation est indiscutable pour le développement de ce type de réseau.

La tendance actuelle pour résoudre ces problèmes dans les réseaux fixes, est réalisée par l'intermédiaire d'un modèle par politiques. Ce modèle [4] utilise un serveur de politiques PDP (Policy Decision Point) qui a pour rôle de prendre les bonnes décisions, les transmettre à ses clients appelés PEP (Policy Enforcement Point) en utilisant le protocole COPS (Commnon Open Policy Service), puis les applique sur ces PEPs. COPS [5] est un protocole de signalisation flexible conçu pour supporter plusieurs types de clients (COPS-RSVP, COPS-PR, COPS-SLS) [6,7,8] dépendamment de la tâche et des décisions politiques qu'il transporte.

L'utilisation d'un tel serveur de politique, permettant d'assurer une gestion par politiques de la sécurité de bout en bout et de la qualité de service, nous semble être comme une idée très intéressante pour les réseaux ad hoc. Il nous a fallût donc adapter ce modèle par politiques aux réseaux ad hoc, car contrairement aux réseaux fixes qui reposent sur une infrastructure et qui contiennent des serveurs dédiés, les réseaux ad hoc ne reposent sur aucune infrastructure et ne sont composés que de nœuds mobiles. Il paraît alors évident que le serveur de politiques devra être supporté par l'un de ces terminaux mobiles.

Nous proposons, donc, dans ce travail d'adapter ce modèle par politiques pour les réseaux ad hoc et plus spécifiquement de proposer un mécanisme permettant l'élection du meilleur terminal pouvant supporter le rôle de serveur de politiques. Cette élection doit prendre en considération plusieurs critères liés aux capacités des terminaux mobiles.

Notre propositions est compatible aux travaux menés dans le cadre du contrôle des réseaux à base de politiques. C'est une solution robuste traitant tous les cas de figures qui peuvent survenir dans un réseau ad hoc (contention entre deux mobiles pour le rôle de serveur, le serveur quitte le réseau, transfert du serveur vers un autre terminal présentant de meilleures capacités, etc.).

Une extension du protocole COPS-SLS a également été nécessaire dans le cadre de l'adaptation proposée. COPS-SLS semble être le type de client le mieux adapté à notre proposition. En effet, ce protocole considère que le PEP peut être supporté par le terminal contrairement aux autres types de clients où le PEP se trouve toujours au niveau des routeurs.

La suite de l'article est structurée de la façon suivante. La section 2 fait une brève introduction des réseaux ad hoc ainsi que des problèmes rencontrés dans ce genre de réseaux. Après une présentation du modèle par politiques pour les réseaux fixes dans la section 3, nous développons notre solution pour l'adaptation de ce modèle pour les réseaux ad hoc. L'extension du protocole COPS-SLS est présentée dans la section 5. Finalement nous présenterons les conclusions et les perspectives de ce travail.

2. Les réseaux ad hoc

L'essor des technologies sans fil offre de nouvelles perspectives dans le domaine des télécommunications. De très nombreux systèmes utilisent déjà ces techniques et connaissent une très forte expansion à l'heure actuelle (notamment la radio téléphonie mobile) mais requièrent une importante infrastructure logistique et matérielle fixe.

Les réseaux sans fil ad hoc, quant à eux, ne nécessitent aucune infrastructure fixe. Ces réseaux sont constitués d'un ensemble d'unités mobiles qui sont libres de se déplacer à leur guise et dont le seul moyen de communication est l'utilisation des interfaces sans fil, sans l'aide d'une infrastructure préexistante ou d'une administration centralisée. L'absence d'un réseau filaire composé de stations de base oblige les nœuds mobiles à intégrer des fonctionnalités qui

n’étaient rempli, jusqu'à présent, que par des nœuds dédiées. En effet, ces nœuds mobiles doivent se comporter comme des routeurs afin de participer à la découverte et à la maintenance des chemins pour les autres hôtes. La gestion de ce routage est assez difficile puisqu’elle consiste à établir une sorte d’architecture molle qui doit tenir compte de la mobilité des stations et de la versatilité du médium physique.

Vu la difficulté de la modélisation de cette architecture, un groupe de travail, appelé MANET (Mobile Ad hoc NETwork) [9], a été crée dans ce sens à L’IETF. Ce groupe de travail œuvre à la normalisation des protocoles de routage ad hoc fonctionnant sous IP. Partant des protocoles de routage de l’IP fixe, MANET en a proposé une extension en tenant compte de la mobilité des nœuds. Hormis les travaux sur les protocoles de routage [10,11], les études liées à la sécurisation des communications et à la garantie de QoS sont restées à un stade embryonnaire. Les techniques à proposer doivent alors tenir compte des caractéristiques particulières de ces réseaux :

- une topologie dynamique et imprévisible qui peut évoluer très rapidement comme le montre la figure 1 ;
- un canal de communication sans fil et donc, des liaisons à débits variables et à bande passante limitée ;
- une autonomie réduite vu que les nœuds fonctionnent souvent avec des batteries et utilisent leurs énergies, en plus de leurs besoins propres, pour router des paquets destinés à d’autres nœuds du réseau.

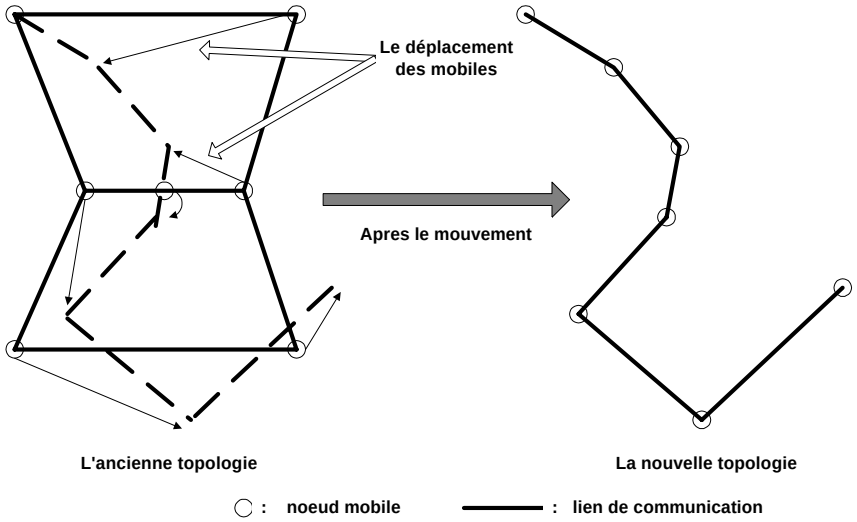

Figure 1. Changement de topologie d’un réseau ad hoc

Un contrôle et une administration adaptés à ces caractéristiques deviennent alors plus que nécessaire pour le bon développement de ce genre de réseaux.

3. Le modèle par politiques

Le terme *politique* [4] étant largement utilisé, il est impératif de donner une définition claire dans le contexte des réseaux. Comme point de départ, le terme politique dénote un règlement unifié pour l’accès aux ressources et aux services réseaux en se basant sur des critères administratifs. La figure 2, dénote différents niveaux auxquels un tel règlement peut être exprimé et exercé. La vue réseaux d’une politique s’exprime en terme de performances de bout en bout, de connectivité et d’états dynamiques du réseau. Cette vue se compose quand à elle de plusieurs vues nodales, auxquelles correspondent les objectifs de la politique et ses

besoins au niveau de différents nœuds. Celles ci, à leurs tours, sont composées de règles politiques, lesquelles doivent être vues comme des injonctions atomiques à travers lesquelles différents nœuds du réseau sont contrôlés. Comme chaque nœuds possède des mécanismes d'allocation de ressources spécifiques, chaque vue nodale doit finalement être translatée en des instructions spécifiques aux dispositifs hardwares du nœud.

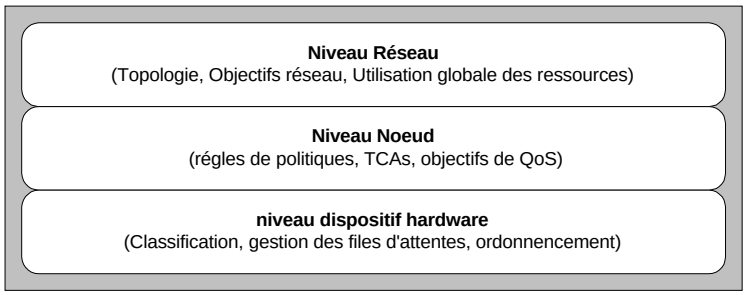

Figure 2. hiérarchie conceptuelle de politique.

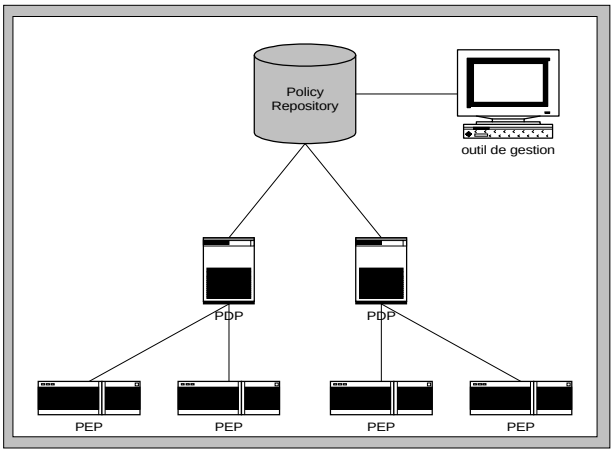

Figure 3. Architecture de contrôle par politique.

La figure 3, montre un model généralisé de l'architecture de contrôle par politique. Cette figure montre les différents composants d'un domaine de politique, que nous allons détailler plus bas.

Un domaine de politique est la portion d'un réseau qui est administrée et gérée par un même ensemble de politiques. Ces politiques sont stockées dans une entité appelée *policy repository*. Le *PEP*[1] est l'élément responsable de l'application et de l'exécution des actions de la politique. C'est également l'élément qui va prendre en charge le traitement des paquets en fonction d'une politique donnée. Le *PDP*[2], quand à lui, a pour tache de déterminer quelle action va être appliquée et à quel paquet. Le PDP interprète les règles des politiques, stockées dans le policy repository, pour un ou plusieurs PEPs. Pour cela le PDP se base sur les conditions courantes du réseau ainsi que sur les informations transmises par les PEPs en utilisant un protocole de signalisation tel que COPS [5]. En effet le protocole COPS semble être le candidat privilégié par l'IETF pour cette tache.

Le PEP peut demander au PDP de prendre des décisions en son nom sur l'occurrence d'un événement (modèle *Outsourcing* [6]) ou faire une demande de configuration (modèle *Provisionning* [7]). Dans le cas des réseaux DiffServ par exemple, le modèle Outsourcing permet de faire un contrôle d'admission au niveau des routeurs de bordures où l'arrivée d'un nouveau flux représente l'évènement. Le modèle Provisionning, quand à lui, permet de

[1] Policy Enforcement Point
[2] Policy Decision Point

configurer les nœuds de bordure et de cœur du réseau afin de fournir une certaine qualité de service. Cette configuration ne nécessite aucun évènement déclencheur au niveau du PEP.

Le protocole COPS [5] est un protocole de *requêtes/réponses* permettant le transport d'informations politiques entre le PDP et le PEP et inversement. Suivant le type d'informations politiques transportées, des extensions spécifiques doivent être définies. Chacune de ces extensions va utiliser l'un des deux modèles présentés plus haut [6,7], voir les deux modèles en même temps [8]. Nous nous intéressons dans ce papier plus particulièrement aux objets et aux messages définis dans l'extension COPS-SLS [8]. En effet, le choix du protocole COPS-SLS est motivé par le fait que ce dernier prend en compte le fait que le PEP puisse être localisé au niveau du terminal et rend donc le terminal responsable du trafic qu'il génère. Nous verrons dans la section 5 que dans le cadre de l'adaptation du modèle par politiques aux réseaux ad hoc, des objets et des messages supplémentaires doivent être rajoutés à COPS-SLS.

4. Adaptation du modèle par politiques aux réseaux ad hoc

Dans cette section, nous allons expliquer comment adapter le modèle par politiques, présenté dans la section précédente, aux réseaux ad hoc et donc expliquer comment introduire un serveur de politique[3] dans un tel réseau. Cette solution va permettre de contrôler le réseau, à base de politiques, sans que cela aille à l'encontre de la définition ad hoc. La principale caractéristique d'un réseau ad hoc, comme nous l'avons vu dans la section 2, est qu'il ne repose sur aucune infrastructure préexistante. Or, le modèle par politiques passe par une certaine centralisation où une machine, voir plusieurs, doit jouer le rôle de serveur de politiques. La question qui se pose alors, vu qu'un réseau ad hoc n'est constitué que d'un ensemble de terminaux mobiles, est "*Quel terminal aura la tache de serveur de politique dans un réseau ad hoc ?*". Le plus avantageux c'est que la machine choisie soit la plus performante possible, car le fait de supporter le serveur de politiques va pénaliser faiblement le terminal si ce dernier est puissant. L'élection d'un des terminaux en tant que serveur de politique se fera par l'intermédiaire de son "*poids*". Le "*poids du terminal*" est l'entité qui va déterminer son degré de capacité. Outre l'affectation d'un poids pour chaque terminal, l'élection du serveur de politiques passe par un certain nombre d'algorithmes distribués que l'on détaillera plus bas. Ces algorithmes doivent prendre en compte la dynamicité des arrivées/départs des terminaux dans le réseau.

4.1. Poids du terminal

Le calcul du poids du terminal (voir figure 4) doit suivre un certain nombre de critères. Ces critères vont permettre de déterminer si le terminal a les ressources nécessaires pour devenir un serveur de politiques ou pas. En effet, le terminal qui va jouer le rôle de serveur de politiques ne doit, en aucun cas, être dédié seulement à cette tâche. L'utilisateur ne se connecte pas pour gérer le réseau mais pour pouvoir communiquer et échanger des informations avec d'autres utilisateurs. Pour cette raison, le choix des critères doit être rigoureux. Le nœud sélectionné doit être multitâche, doit avoir une forte puissance de calcul (processeur, mémoire RAM, etc.), doit être bien positionné géographiquement dans le réseau ad hoc, et bien d'autres critères encore. Le choix de ces critères dépend de contraintes d'implémentation et est hors du champs de ce papier. Cependant, vu la variété des critères possibles, nous les avons séparé en deux grandes catégories :

1) **Les critères liés à des conditions nécessaires :** Cette catégorie contient les critères indispensables au terminal pour lui permettre de supporter un serveur de politiques (exp. Multitâche). En effet, si un de ces critères n'est pas assuré par le terminal, celui

[3] Nous entendons ici par *serveur de politiques* le regroupement du PDP et du Policy Repository

ci ne pourra pas être élu comme serveur de politiques vu le risque de pénaliser l'utilisateur. Une machine monotâche, par exemple, ne pourra exécuter que l'application serveur de politique si elle est élu. Dans ce cas, l'utilisateur ne pourra plus utiliser sa machine. Dans le cas où le terminal ne supporte pas un des critères liés aux conditions nécessaires, la valeur nulle sera affectée à son poids.

2) **Les critères liés à des aspects de performances :** Cette catégorie, comprend les critères qui vont servir à départager les terminaux éligibles au poste de serveur de politiques (c-à-d, les terminaux qui vérifient tous les critères de la première catégorie). Ces critères sont liés aux performances de la machine (exp. RAM, niveau de batterie, processeur,..). Le poids de ces nœuds éligibles sera calculé par la formule suivante :

$$Poids = \frac{1}{\sum_{i=1}^{N} a_i} \sum_{i=1}^{N} a_i f_i(x) \qquad (1)$$

où N est le nombre de critères, a_i est le poids du critère dont la valeur dépend de l'importance du critère et $f_i(x)$ est une fonction qui dépend du critère.

Si (toutes les conditions nécessaires = = *Vraie*) *alors*
{
/* calcul du Poids du terminal */
$Poids = \frac{1}{\sum_{i=1}^{N} a_i} \sum_{i=1}^{N} a_i f_i(x)$; // x est la valeur du critère
}
Sinon
{
Poids = 0 ; //Cas ou le terminal ne peut devenir un PDP
}

Figure 4. Algorithme de "calcul du poids".

4.2. Connexion au réseau

Dés la connexion d'un nouveau terminal au réseau, son poids est calculé. Dans un premier temps, le terminal va essayer de détecter la présence ou non d'un serveur de politiques. La détection peut se faire par l'intermédiaire du protocole SLP[4] (Service Locator Protocol) ou SDP[5] (Service Discovery Protocol). Le terminal se trouve alors confronté à l'un des cas de figures suivants :

- Un serveur est déjà présent dans le réseau ad hoc (*SExist=Vrai*)
- Aucun serveur n'est présent ni en train de se déclarer (*SExist=Faux, SPossible=Faux*)
- Un serveur est entrain de se déclarer (*SExist=Faux, SPossible=Vrai*)

1) Cas où le terminal détecte la présence d'un serveur : Dans ce cas, le terminal va directement se connecter au serveur en tant que client (PEP). Le terminal va alors suivre la procédure de connexion décrite par le protocole COPS. Une fois que la connexion est établie avec le serveur, le terminal lui envoie alors son poids. Si ce poids est non nulle, le serveur va

[4] SLP est un protocole utilisé par COPS pour localiser le serveur de politique.
[5] SDP est un protocole utilisé dans les réseaux ad hoc pour permettre aux applications de découvrir les services offerts par les autres terminaux.

le mémoriser. Les poids mémorisés par le serveur seront utilisés ultérieurement pour un éventuel changement de serveur. Ce dernier point sera détaillé dans les sections suivantes.

2) Si aucun serveur n'est présent ni en train de se déclarer : Le terminal se réfère à son poids pour déterminer s'il peut ou non supporter le rôle de serveur. Dans le cas où le terminal aurait les capacités nécessaires (*poids* $\neq$ *0*), il déclenche la procédure de mise en place du serveur. Dans le cas contraire (*poids=0*), il attend qu'un serveur se déclare pour se connecter en tant que PEP.

3) Un serveur est en train de se déclarer : Ce troisième cas a été ajouté pour éviter que plusieurs serveurs ne se déclarent en même temps. En effet, la procédure de déclaration d'un serveur peut ne pas être atomique. Pour pallier à ce problème, le terminal qui veut se déclarer comme serveur de politique devra indiquer, à tout terminal qui essaye de détecter la présence d'un serveur de politique, qu'il est entrain de se déclarer (*SPossible=Vrai*). Le terminal qui sera informé ne pourra plus se déclarer comme serveur et devra attendre que le serveur soit mis en place pour s'y connecter en tant que PEP. Il est à noter que si le terminal qui est en train de se déclarer comme serveur échoue pour une raison quelconque, la variable *SPossible*[6] reprendra à la valeur "*Faux*". Dans ce cas, un des clients en attente lancera la procédure de mise en place du serveur comme décrit plus haut.

La procédure complète de connexion au réseau est décrite par l'algorithme de la figure 5.

```
Rechercher de la présence d'un serveur ;
Si  (SPossible  = = Vrai) alors
{
        Attendre (X) ;      // Temporisation de X secondes.
        Re-déclenchement de l'algorithme "nouvel adhérent" ;
}

Si (SExist = = Vrai) alors
{
        La machine se connecte en tant que PEP;
        Envoi du Poids de la machine au PDP;
}
Sinon
{
        Si ( Poids  ≠ 0)
        {
                SPossible = Vrai ;
                PolicyServer();    // déclenchement de la procedure pour devenir Serveur.
        }
        Sinon
        {
                Attendre (X) ;      // Temporisation de X secondes.
                Re-déclenchement de l'algorithme "nouvel adhérent" ;
        }
}
```

Figure 5. Algorithme "nouvel adhèrent".

4.3. Durée de vie d'un serveur de politiques

Dans cette section, on va présenter les différents cas de figures qui peuvent survenir lors de la durée de vie d'un serveur de politiques dans un réseau ad hoc. Trois cas se présentent :

[6] Les variables *SExist* et *SPossible* sont mise à jours par les algorithmes SLP et SDP.

- Le serveur quitte le réseau ;
- Changement de serveur pour une machine ayant de meilleures performances ;
- Le serveur disparaît de façon brusque.

4.3.1. Le serveur quitte le réseau

C'est le cas où le terminal supportant le serveur s'apprête à quitter le réseau[7]. Cette opération doit être précédée par un transfert du serveur vers un autre terminal afin d'éviter que le réseau ne se retrouve sans serveur et donc sans contrôle. Cette opération n'est bien sûre possible que s'il y'a un terminal éligible en tant que serveur (*poids* $\neq 0$). Comme le montre l'algorithme de la figure 6, le transfert se fait comme suit :

1- Avant de quitter le réseau, le serveur consulte la table dans laquelle il a mémorisé les poids de tous ses clients.

2- Il choisit le client qui a le poids le plus élevé pour lui envoyer un message de décision qui contient la commande *START*. Cette commande oblige le terminal à démarrer le serveur dès qu'il reçoit le message de fermeture de client.

3- Le serveur attend de recevoir un message de rapport du client, afin de garantir la présence d'un nouveau serveur. Si le serveur reçoit ce message il envoie un message de fermeture de client à tous ses PEPs et il les dirigent vers le nouveau serveur sinon il n'y aura pas de transfert.

```
Si (ServeurQuitte = = Vrai)
{
        Envoie d'un message contenant la commande START au PEP qui a le
        meilleur poids;
          Si (Rapport_arrivé = = Vraie)
                {
                        Envoie un message de fermeture de client contenant
                        l'adresse du PDP alternatif à tous ses PEPs ;
                }
          Sinon
                {
                        SExist = Faux ;
                }
        Fermeture du PDP ;
}
```

Figure 6. Algorithme "serveur quitte".

4.3.2. Changement de serveur

Lorsque le serveur détecte un terminal offrant de meilleures performances, il effectue alors un transfert du serveur vers ce terminal. Cette détection se fait périodiquement en comparant son propre poids avec les différents poids qu'il a mémorisé. Ce transfert s'effectue seulement si la condition suivante est vérifiée :

$$Taux = \frac{\text{Poids du terminal le plus élevé}}{\text{Poids du serveur actuel}} > \alpha \qquad (2)$$

où α est un seuil au delà duquel un transfert de serveur est préférable.

[7] L'utilisateur termine son travail et veut éteindre sa machine ou bien le signal devient trop faible.

Pour éviter toute instabilité du système qui pourrait être causé par des changements trop fréquents de serveur (à chaque fois qu'il y a un terminal meilleur que le serveur actuel on fait un transfert), un temporisateur a été défini. Comme le montre l'algorithme de la figure 7, tant que ce temporisateur n'a pas expiré, aucun transfert n'est possible. A l'expiration du temporisateur, le serveur actuel recherche si un autre terminal vérifie la condition (2). Si c'est le cas, le nouveau terminal prend le relais.

```
Attendre (Y);      // Y : valeur du temporisateur
Taux = (Poids du terminal le plus élevé) / (Poids du serveur actuel) ;
Si (Taux > α )
        {
                Exécution de l'algorithme "serveur quitte" ;
        }
```

Figure 7. Algorithme "changement de serveur"

4.3.3. Le serveur disparaît

C'est le cas où le serveur disparaît brusquement du réseau[8]. Les PEPs vont détecter, par l'intermédiaire des messages *Keep Alive* du protocole COPS, la disparition du PDP. Dès qu'un PEP détecte la disparition du serveur, la variable *SExist* est mise à faux. Tous les PEPs enclenchent alors à nouveau la procédure de reconnexion au réseau (voir l'algorithme de la figure 8).

```
Si (SExist = = Faux) alors
{
        Exécution de l'algorithme "nouvel Adhérent" ;
}
```

Figure 8. Algorithme "*serveur disparaît*"

Après avoir décrit notre proposition, qui consiste à élire un des terminaux du réseau ad hoc pour qu'il devienne serveur de politiques, nous allons dans la section suivante détailler les modifications apportées à COPS-SLS pour qu'il puisse supporter cette élection.

5. Extension du protocole COPS-SLS

Le type de client COPS-SLS [8] semble convenir le mieux à l'adaptation du modèle par politiques que l'on désire effectuer. En effet, ce client présente de nombreux avantages par rapport aux autres types de client. Le premier de ces avantages est qu'il considère que le PEP peut être localisé au niveau d'un terminal, contrairement à COPS-RSVP ou COPS-PR où le PEP se trouve toujours au niveau d'un routeur. Le second avantage, c'est qu'il permet la mise en place d'une négociation entre le PEP et le serveur. En effet, il permet au serveur de proposer d'autres services lors de la négociation. Dans notre cas, nous pourrons alors considérer l'envoi du poids du terminal comme étant une requête et l'envoi de la commande *START* (cf. section 4.3.1) comme étant un service proposé par le PDP.

Afin que notre proposition puisse fonctionner avec COPS-SLS, nous proposons d'ajouter deux nouveaux objets (voir figure 9) :

- Le premier objet, appelé **PoidsT**, permet de transporter le poids du terminal du client vers le serveur ;
- Le deuxième objet, appelé **CommandeS,** transporte la commande *START* du PDP au PEP pour obliger le PEP à devenir serveur.

[8] Le terminal supportant le serveur plante.

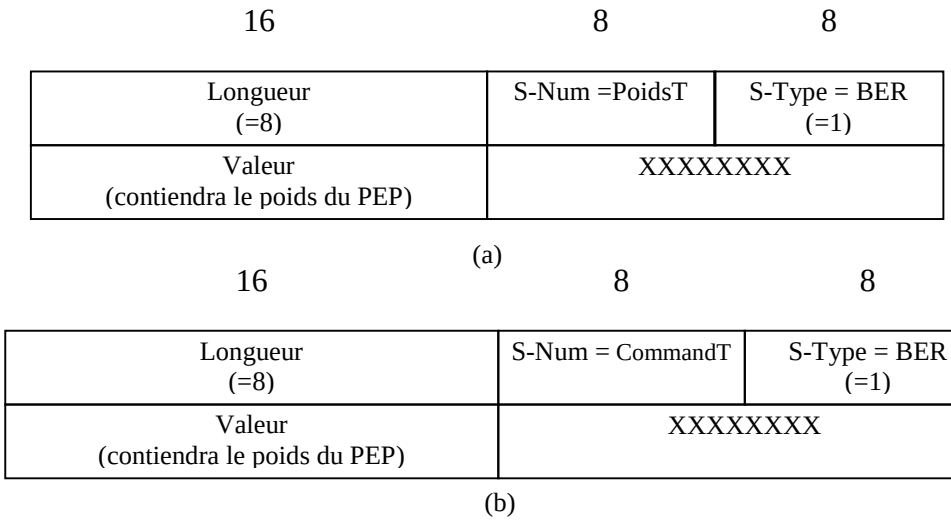

Figure 9. les objets (a) **PoidsT** et (b) **CommandeS**.

5.1. L'objet PoidsT

La première chose que va faire un PEP, après avoir établit une connexion avec le serveur, est d'envoyer un message d'ouverture de client avec le type de client COPS-SLS. Comme le montre la figure 10, le serveur répond dans le cas positif par un message d'acceptation de client, sinon par un message de fermeture de client. Ceci est le comportement classique de COPS.

Une fois que le PEP reçoit le message d'acceptation de client, il envoie un message de requête de configuration et va attendre la décision du PDP. En plus des informations que le message de requête contient, on envoie aussi le poids du terminal qui sera mémorisé par le PDP. L'objet **PoidsT** sera alors encapsulé dans l'objet optionnel *Named ClientSI* (voir figure 10).

5.2. L'objet CommandeS

Lorsque le serveur détecte qu'il existe un terminal qui possède de meilleures capacités que lui, il lui envoie un message de décision pour l'informer qu'il va devenir serveur. Ce message ne contiendra pas de décision, mais plutôt l'objet optionnel *Named Decision Data* qui encapsulera l'objet **CommandS** définit plus haut. Ce message va obliger le PEP à démarrer la procédure de mise en place du serveur. Par la suite le PDP envoi à tous ses PEPs un message de fermeture de client qui contiendra l'objet *PDPredirect* pour les rediriger vers le nouveau serveur. Avant d'envoyer ce message de fermeture, le PDP attend de recevoir un message de rapport du client désigné.

La même procédure est effectuée lorsque le serveur veut quitter le réseau et qu'un autre terminal est éligible pour prendre le rôle de serveur de politiques. Dans le cas ou aucun terminal n'est éligible, le message de fermeture de client ne contiendra pas l'objet *PDPredirect*.

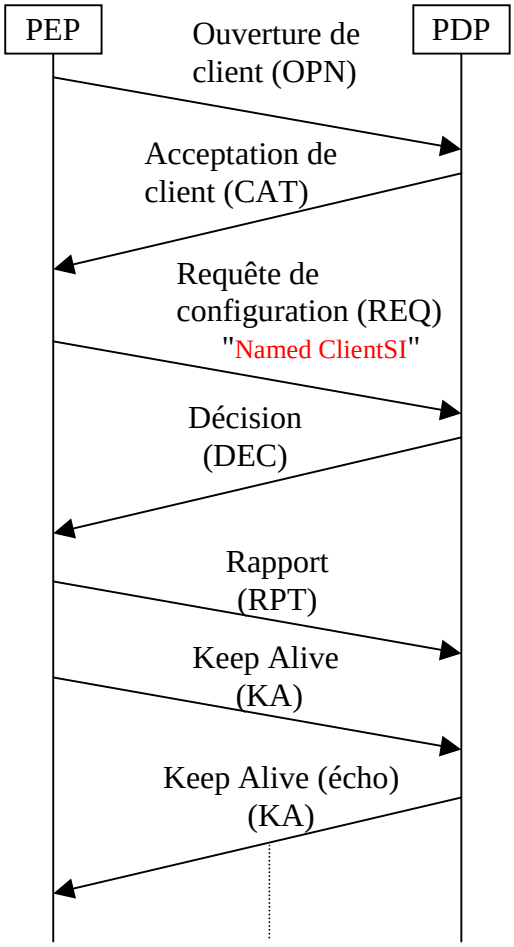

Figure 10. Etablissement de la connexion avec échange du poids.

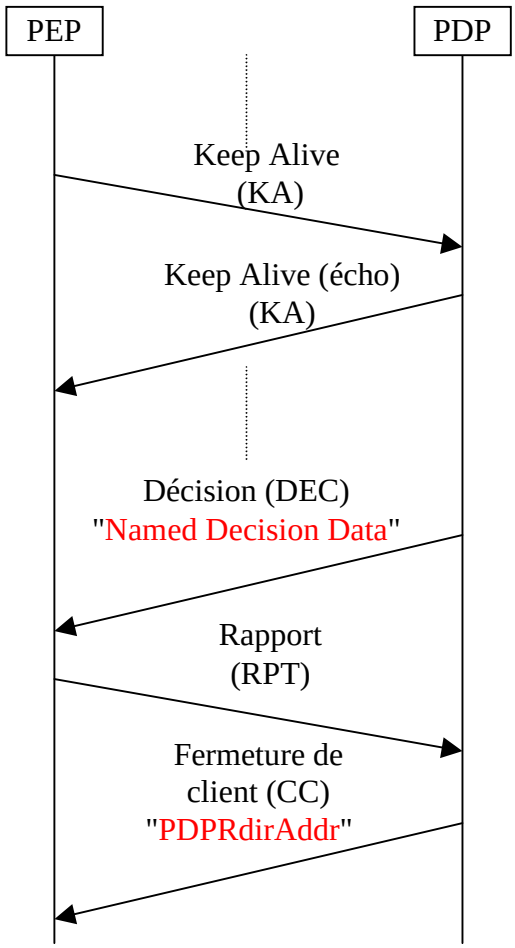

Figure 11. Transfert du PDP.

6. Conclusion et perspectives

Pour pallier aux problèmes de sécurisation des communications et de garantie de qualité de service de bout en bout, la tendance actuelle repose sur l'utilisation d'un contrôle de réseaux à base de politiques. Les premiers résultats apportés par cette approche dans le cadre des réseaux fixes semblent être très prometteurs. Nous proposons, dans ce travail, d'appliquer ce même concept aux réseaux ad hoc. Mais, vu que les réseaux ad hoc ne reposent sur aucune infrastructure préexistante et que tous les nœuds du réseaux peuvent être mobiles une adaptation de ce modèle de contrôle a été nécessaire. Cette adaptation passe par l'élection d'un des terminaux du réseau pour jouer le rôle de serveur de politiques. La méthode proposée permet de mettre en place ce serveur de politiques en se basant sur le poids des nœuds du réseaux. Ce poids permet, dans un premier temps, d'éliminer tout terminal qui ne possède pas les ressources nécessaires pour supporter le serveur. Dans un second temps, le poids permet de choisir le terminal le mieux adapté pour le support de la charge d'un serveur de politiques. Afin de rendre notre proposition plus robuste, nous avons traité le cas de la contention entre deux machines pour le rôle de serveur de politiques ainsi que le cas de l'instabilité causé par des transferts trop fréquents du serveur de politiques. La solution proposée est flexible et s'accorde bien à la nature dynamique du réseau. Dans le cadre de notre étude une extension du protocole COPS-SLS a également été nécessaire.

Afin de compléter ce travail, un certain nombre de points tels que la sécurisation du serveur de politiques ainsi que sa transparence par rapport aux utilisateurs doivent être étudiés et

feront l'objet de travaux futures. La proposition de politiques propres à la garantie de QoS de bout en bout et à la sécurisation des communications dans les réseaux ad hoc fait également partie des perspectives de cette étude.

7. Bibliographie

[1] "LAN MAN Standards of the IEEE Computer Society. Wireless LAN medium access control (MAC) and physical layer (PHY) specification. IEEE Standard 802.11", 1997.

[2] Chaudet C., "Qualité de service et réseaux ad-hoc - un état de l'art", *Rapport de recherche INRIA, RR-4325, Novembre 2001*.

[3] Fluhrer, S., Mantin, I., et Shamir, A., "Weaknesses in the Key Scheduling Algortihm of RC4", *Eight Annual Workshop on Selected Areas in Cryptography, Août 2001*.

[4] Rajan R., Verma D., Kamat S., Felstaine E., and Herzog S., « A policy framework for integrated and differentiated services in the Internet », *IEEE Network Magazine, vol. 13, no. 5, pp. 36-41, September / Octobre 1999.*

[5] Durham D., et al., « The COPS (Common Open Policy Service) Protocol », *RFC 2748, Janvier 2000.*

[6] Herzog S., et al., « COPS usage for RSVP », *RFC 2749, January 2000.*

[7] Chan K., et al., « COPS Usage for Policy Provisioning (COPS-PR) », *RFC 3084, Mars 2001*.

[8] Nguyen, T.M.T., Boukhatem, N., El Mghazli, Y., Pujolle, G., « COPS Usage for SLS negotiation », *Draft IETF standard, Salt Lake City, Novembre 2001*.

http://www.ietf.org/internet-drafts/draft-nguyen-rap-cops-sls-01.txt

[9] Mobile Ad-hoc NETworks (MANET), groupe de travail de l'IETF, http://www.ietf.org/html.charters/manet-charter.html.

[10] Royer, E. M., et Toh, C-K., "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks", *IEEE Personal Communications, Avril 1999.*

[11] Samir, R. D., C. Robert, C., Jiangtao, Y., et Rimli, S., "Comparative performance evaluation of routing protocols for mobile, ad hoc networks", *IEEE the Seventh International Conference on Computer Communications and Networks (IC3N '98), Octobre 1998.*