GRES 2006

7ème Colloque Francophone
de Gestion de Réseaux et de Services

L'autonomie dans Les Réseaux et Les Services

Sous la direction de

Francine Krief

Actes du 7ème Colloque Francophone
de Gestion de Réseaux et de Services

du 9 au 12 mai 2006
au LaBRI, Université Bordeaux-1
France

GRES 2006

L'autonomie dans Les Réseaux et Les Services

Sous la direction de

Francine Krief

Présidente du colloque

Francine Krief (LaBRI, ENSEIRB / Université de Bordeaux-1, France)

Comité Scientifique

- Armen Aghasaryan (Alcatel, France)
- Nazim Agoulmine (LRSM (Université d'Evry Val d'Esssonne), France)
- Toufik Ahmed (LaBRI, France)
- Spiros Arsenis (Alcatel, France)
- Otto Carlos Muniz Bandeira Duarte (Universidade Federal do Rio de Janeiro, Brésil)
- François Barrère (IRIT, France)
- Younes Bennani (LIPN, Univ. Paris-13, France)
- Abderrahim Benslimane (LIA, Université d'Avignon et des Pays de Vaucluse)
- Abdelmalek Benzekri (IRIT, France)
- Kamel Barkaoui (CNAM-Paris, France)
- Mounir Boukaddoum (UQÀM-Montréal, Canada)
- Raouf Boutaba (University of Waterloo, Canada)
- Antoine Boutignon (Cegetel, France)
- Sandrine Calomme (Université de Liège, Belgique)
- Richard Castanet (LaBRI, France)
- Ken Chen (Université Paris 13, France)
- Prosper Chemouil (FTR&D, France)
- Omar Cherkaoui (UQÀM-Montréal, Canada)

- Jean-Pierre Claudé (PRISM, France)
- Andre Cotton (Thalesgroup, France)
- Xavier Delord (LaBRI, France)
- Gilles Desoblin (Alcatel, France)
- Thierry Desprats (IRIT, France)
- Olivier Festor (INRIA Lorraine, France)
- Monique Gibeaux (Bouygues Telecom, France)
- Alejandro Guerra Hernandez (Universidad Veracruzana, Mexique)
- Marie-Pierre Gervais (Lip6 et Paris X, France)
- Abdelhakim Hafid (University of Montreal, Canada)
- Nada Kabalan (Liban)
- Farouk Kamoun (ENSI, Tunisie)
- Ahmed Karmouch (Université d'Ottawa, Canada)
- Daniel Kofman (ENST- Paris, France)
- Jacques Labetoulle (Groupe des Ecoles des Télécommunications, France)
- Adina Magda Florea (University Politehnica of Bucharest, Romanie)
- José Marcos Silva Nogueira (Universidade Federal de Minas Gerais, Brésil)
- Joberto SB Martins (JSMNET, Brésil)
- Zoubir Mammeri (IRIT , France)
- Oliveira Mauro (CEFET-CE, Brésil)
- Jose Neuman de Souza (Federal University of Ceara, Brésil)
- Fetah Ouzzani (Qosmic, France)
- Harry Perros (NC State University, USA)
- Samuel Pierre (École Polytechnique de Montréal, Canada)
- Ramon Puigjaner (Universitat de les Illes Balears, Espagne)

- Guy Pujolle (Laboratoire Lip6, France)
- Jose Rezende (Universidade Federal do Rio de Janeiro, Brésil)
- Nicolas Rouhana (University Saint-Joseph, Liban)
- Mikael Salaun (FranceTelecom, France)
- Ahmed Serhrouchni (ENST-Paris, France)
- Noëmie Simoni (ENST-Paris, France)
- Michelle Sibilla (IRIT, France)
- Henry Soldano (LIPN, Paris-13, France)
- Sami Tabbane (SupCom, Tunisie)
- Carlos Becker Westphall (UFSC, Brésil)
- Simon Znaty (Efort, France)

Comité d'organisation

- Toufik Ahmed (LaBRI, France)
- Badr Benmammar (LaBRI, France)
- Véronique Bogati (Labri-transfert, cellule de transfert et valorisation)
- Lahcene Dehni (LIPN, France)
- Xavier Delord (LaBRI, France)
- Ismail Djama (LaBRI, France)
- Zeina Jrad (LIPN, France)
- Nader Mbarek (LaBRI, France)
- Mubashar Mushtaq (LaBRI, France)
- Hassine Moungla (LIPN, France)

Table des matières

Avant-propos

Le Colloque Francophone sur la Gestion de Réseaux Et de Services est un lieu de rencontre et d'échanges entre scientifiques et chercheurs francophones venant du monde entier. Ce Colloque est actuellement à sa septième édition. Il a eu lieu à PARIS (1995), RENNES (1997), MONTRÉAL (1999), MARRAKECH (2001), FORTALEZA (2003), LUCHON (2005) et se tient actuellement à Bordeaux du 9 au 12 mai 2006.

L'objectif de GRES est de permettre à la communauté francophone des chercheurs, scientifiques et industriels ayant un intérêt pour la gestion de réseaux et services de faire régulièrement le point sur les recherches et les expérimentations en cours de développement dans ce domaine. Cette année, ce colloque comprend 7 tutoriaux et 5 sessions de conférences de travaux et projets de recherche et développement.

Le thème retenu en 2006 est « La gestion autonome des réseaux et services ». En effet, la complexité croissante des fonctions de gestion, la difficulté à trouver des personnes très fortement qualifiées pour maîtriser cette complexité et le besoin de maîtrise des coûts liés à l'activité de contrôle et de gestion des réseaux et services favorisent l'émergence d'un nouveau paradigme que l'on nomme « la gestion autonome ». Selon cette vision, le réseau est capable de se configurer automatiquement (auto-configuration), de chercher en permanence à améliorer ses performances (auto-optimisation), de détecter, de diagnostiquer, et de réparer les problèmes d'équipement et de logiciel (auto-rétablissement) et de se protéger des attaques ou des cascades de défaillances (auto-protection).

Toutefois, cette vision de la gestion n'est qu'une finalité opérationnelle qui n'apporte pas de solution quant aux moyens à mettre en œuvre pour atteindre cette autonomie de gestion. Ces actes apportent une première série de réponses.

Francine Krief
Présidente du GRES 2006

SESSION 1

LA QUALITE DE SERVICE

GXLA Un Langage de Spécification des SLA

Badis TEBBANI, Issam AIB, Guy PUJOLLE

Équipe PHARE Laboratoire LIP6
Université Pierre et Marie Curie
08 rue du Capitaine Scott
75015 Paris.
{badis.tebbani, issam.aib, guy.pujolle}@lip6.fr

RESUME *Dans ce travail, nous proposons le langage GXLA pour la spécification basée XML des contrats de niveau de service. Le GXLA permet la formalisation du contrat et la négociation du niveau de service, d'une part, et sa traduction en performance du système d'autre part. Le GXLA représente la spécification sous forme de langage du modèle d'information GSLA (Generalized Service Level Agreement) que nous avons présenté dans un travail antérieur. Nous présentons les avantages de notre langage et nous le comparons à d'autres propositions de spécification des SLA qui se trouvent dans la littérature. Un cas d'étude est utilisé pour tester la faisabilité du GXLA.*

ABSTRACT *In this work, we propose GXLA, a language for the specification of Service Level Agreement (SLA). GXLA represents the implementation of the Generalized Service Level Agreement (GSLA) information model we proposed in a previous work. GXLA supports multi-party service relationships through a role-based mechanism. It is intended to catch up the complex nature of service interactivity in the broader range of SLA modelling of all sorts of it business relationships. The GXLA language is based on XML; it is defined as an xml schema. GXLA can be used by both service providers and service customers in order to configure their respective systems. Each party can use its own independent SLA interpretation and deployment technique to enforce the role it has to play in the contract. Finally, we illustrate our proposition through a use case.*

MOTS-CLES: *Contrat de Niveau de Service; Gestion par Politiques; Spécification des SLA.*

KEY-WORDS: *Service Level Agreement; Policy Based Management; SLA specification.*

1. Introduction

La gestion des réseaux par politiques (*Policy-based Network Management*) concerne l'utilisation de règles pour gérer la configuration et le comportement d'une ou plusieurs entités dans un système informatique. Une règle fait correspondre un ensemble d'actions à un ensemble de conditions : les conditions sont évaluées afin de déterminer s'il faut appliquer les actions. L'évaluation des conditions est déclenchée grâce à un ensemble d'événements.

Les contrats de niveau de services (SLA) sont un moyen très important de gestion dans un contexte de gestion orientée service. Ils s'imposent jour après jour, et deviennent de plus en plus indispensables pour l'optimisation des ressources et l'automatisation. Les réseaux informatiques où prime la collaboration et les échanges de services doivent adopter la gestion par politiques pour réagir rapidement au changement de l'environnement sans l'intervention humaine. Actuellement il n'existe pas de standard de spécification des SLA. Dans cette optique, il y a beaucoup de propositions qui sont plus ou moins adaptées. La spécification doit être générique pour s'appliquer à tous les scénarios possibles et extensible pour pouvoir ajouter des paramètres spécifiques, ou définir de nouveaux paramètres. On note, dans ce champ de recherche, trois niveaux d'études qui se chevauchent et qui sont très complémentaires. La réussite d'un système PBNM repose sur ces trois solutions : (i) La représentation et l'échange des configurations matérielles (NetConf, 2005)(Boutaba et al., 2002), où l'accent est mis sur la performance et le passage à l'échelle dans un système informatique. Généralement, on suppose que la base de politiques est déjà remplie en respectant les contrats SLA. (ii) La représentation des politiques, la vérification de ces politiques, leur validation, détection/résolution des conflits et leur orchestration. Ici, on chevauche plus avec la configuration matérielle et on propose quelques outils pour introduire les politiques (Bajaj et al., 2004)(James Skene D et al., 2003). (iii) La modélisation du monde extérieur au système, les relations commerciales concernant le système, la façon avec laquelle est négocié ou demandé un produit (service), les paramètres qui décrivent chaque service (Aib I et al., 2004)(Heiko et al., 2003). Là, il est essentiel de disposer d'une représentation précise de ce monde, afin que le produit soit bien défini et compris par tous, aussi bien au niveau affaire (*Business)* qu'au niveau Système. Pour cela, il faut trois éléments : un modèle, un langage et un outil permettant d'établir la correspondance entre les règles du niveau Business et les politiques régissant l'utilisation des ressources des différentes entités.

Cet article est organisé comme suit : dans la prochaine section, on examinera les modèles d'informations des SLA existants ; les travaux existant sur la modélisation des politiques seront présentés dans la troisième section ; notre modèle d'information *GSLA* et sa spécification GXLA seront expliqués dans la quatrième section ; la cinquième section montrera un exemple d'exécution pour tester notre spécification et nous conclurons par quelques perspectives de notre travail.

2. Travaux sur la Spécification des SLA

2.1 *WSLA Web Service Level Agreement IBM2003*

WSLA (Heiko et al., 2003) est avant tout un cadre de travail (framework) pour les SLA. C'est un bon cas d'étude dans lequel sont expliqués les démarches et le déroulement de la conception d'un langage de spécification des SLA. WSLA propose une spécification basée XML. Dans le *WSLA*, la classe SLA contient trois classes : la première décrit les parties impliquées ; la deuxième contient une ou plusieurs descriptions du service et la dernière décrit les engagements.

Deux types de parties peuvent être impliqués dans le service : (i) Partie signataire du SLA concernant le fournisseur du service et le client ; (ii) Partie des sous-traitants (*third-party*). La classe *Parties* a deux signataires et peut avoir plusieurs sous-traitants (0..n). Notre langage *GXLA* élargit la notion du third-party vers celle du multi-partie.

La *description du service* contient une ou plusieurs opérations qui définissent un ou plusieurs paramètres. A leur tour, les Paramètres sont reliés à une métrique. La métrique est soit reliée à une directive indiquant comment elle doit être mesurée, soit elle est reliée à une fonction. Dans ce cas, la métrique peut être la combinaison d'une ou plusieurs autres sous-métriques.

Enfin, la classe *Engagement* contient deux classes d'obligation. La première concerne un ou plusieurs SLO qui garantissent un état du SLA pendant une période déterminée. La seconde représente les actions promises dans le SLA lors d'événements particuliers. Le *WSLA* insiste sur le processus de mesure des métriques et articule sa spécification pour permettre aux fournisseurs de services (ou client) de sous-traiter ce processus à une entreprise tierce. WSLA se base sur les descripteurs de services Web WSDL, SOAP, et UDDI, qui facilitent la description du service et son indexation.

2.2 *SLAng*

C'est un langage des SLA en cours de développement, en tant qu'élément du projet de TAPAS à UCL (Dan Richard A et al, 2002). C'est un langage basé XML et qui repose sur les langages de description des services Web (WSDL) et les applications serveurs (J2EE, CORBA).

SLAng définit sept types de SLA. D'abord, il divise les SLA en deux grandes catégories : (1) *SLA Horizontal* : le contrat entre deux entités égales (du même niveau de l'architecture), eg. Deux applications. (2) *SLA Vertical* : le contrat entre deux entités dans des couches différentes, eg. La couche réseaux avec l'application. *SLAng* introduit la notion de responsabilité Client, responsabilité Fournisseur et la responsabilité Mutuelle (eg. pénalités). Chacune d'elles décrit les obligations de chaque partie. La spécification *SLAng* est en XML. Notre *GXLA* étend cette notion dans le rôle et définit la responsabilité ou les garanties pour un service indépendamment, pour pouvoir

l'affecter ensuite à une ou plusieurs parties. *SLAng* conçoit le SLA comme un contrat entre strictement un client et un fournisseur sans prendre en compte les autres acteurs dans le réseau. Ceci limite sa généralité.

3. Travaux sur la Spécification des Politiques

3.1 *PONDER Imperial College, London*

Le projet *Ponder* a commencé en 1992 dans Imperial College, London. Il offre tout un cadre de travail qui permet de spécifier des droits d'accès et des politiques de gestion avec un contrôle d'accès basé rôle (Dulay Net al., 2001)(Lupu E et al., 2001). C'est un langage déclaratif, orienté objet, basé XML et supporte le typage et la combinaison des politiques. *Ponder* considère une politique comme le choix de paramétrage d'un composant réseau ou l'autorisation d'accès selon des conditions. Il définit plusieurs types de politiques et il s'articule sur le paradigme ECA (Evénement, Condition, Action) (*On Event If Condition Then Action*).

Ponder définit quatre types de politiques : (1) *Authorizations* : comportent essentiellement des règles de contrôle et de droits d'accès, en précisant qui est autorisé ou interdit dans une activité ou par rapport à un élément du système. (2) *Obligations* : spécifient quelles sont les activités que doit tenir une partie, leur invocation (déclenchement) est principalement contrôlée par des événements. (3) *Refrains* : déterminent quelles sont les actions que chaque partie ne doit pas tenir, ce sont des politiques de non autorisation. (4) *Delegations* : spécifient quelles sont les actions que peut déléguer une partie à une autre. Ces quatre types sont les bases de toutes les politiques, *Ponder* définit aussi quatre autres types de composition de politiques pour grouper les politiques appartenant à un même domaine pour simplifier la spécification : *groups*, *roles*, *relationships* et *management structure*. *Ponder* définit également les limites de l'applicabilité de la politique et il introduit la notion de *Domains* pour grouper les politiques qui concernent, le même sujet ou la même région géographique.

Exemple1 :

inst auth+ VideoConf1{
subject /Agroup + NYgroup;
target USAStaff-NYgroup;
action VideoConf(BW, Priority);
when Time.between("1600","1800")
and(Priority>2);}
	Les membres de Agroup et de Bgroup peuvent réaliser une vidéoconférence avec le personnel basé aux Etats-Unis sauf avec le groupe de New York. La contrainte de la politique est ici composée. La contrainte de temps limite le service pour ne s'appliquer qu'entre 4:00pm et 6:00pm et la contrainte d'action indique que la politique est valide seulement si le paramètre prioritaire (2ème paramètre) est plus grand que 2.

Dans *Ponder,* les actions sont soit des fichiers exécutables, soit des scripts externes stockés dans des domaines précis. L'accès à ces domaines est contrôlé par les politiques d'*Authorization*.

Enfin, la spécification *Ponder* est un très bon candidat pour la mise en oeuvre des SLA. Mais, elle doit élever son niveau d'abstraction pour prendre en compte les besoins applicatifs et toute la structure du SLA. Le travail sur Ponder est egalement arrêté depuis deux ans et nous ne voyons pas bien comment il évoluera.

3.2 *WS-POLICY (IBM, Microsoft et all. 2003)*

WS-Policy (Bajaj S et al., 2004) est un cadre de travail qui définit les structures de base pour la description et la communication des politiques des services Web. Il forme une grammaire flexible et extensible pour exprimer les besoins, les capacités et les préférences d'un service Web dans un format XML. Les politiques sont définies dans *WS-Policy* comme une collection d'« alternatives ». L'alternative est une collection d'assertions. L'assertion est l'unité de base des politiques dans *WS-Policy*. Chaque assertion appartient à un domaine spécifique. *WS-Policy* définit les domaines d'application des politiques suivants : *Security, Privacy*, *Application priorities*, *User account priorities* et *Traffic control*. Les assertions peuvent être regroupées ou combinées grâce aux opérateurs <wsp:All>, <wsp:ExactlyOne> et l'attribut de type « Optional ».

WS-Policy ne définit pas la manière dont les politiques seront interfacées avec le service Web, tandis qu'elle laisse le champ libre à d'autres spécifications pour définir la manière dont sont attachées les politiques avec les mécanismes et les ressources du service Web pour une meilleure adaptabilité. Par exemple, *WS-PolicyAttachement* est une spécification qui définit l'association de *WS-Policy* avec WSDL et UDDI.

L'idée principale de *WS-Policy* est d'offrir une grammaire pour décrire les politiques (préférences, capacités…) de chaque service Web et ensuite, de pouvoir échanger ces informations et de comprendre leur sémantique. *WS-Policy* s'intègre dans des messages SOAP, celui-ci étant extensible pour supporter des nouveaux types de messages. *WS-Policy* est la base de plusieurs cadres de travail (frameworks) : *Web Services Policy Attachment*, *Web Services Reliable Messaging*, *Web Services Metadata Exchange*, *Web Services Security Policy*, etc., qui forment ensemble la méta-spécification des politiques des services Web.

Enfin, *WS-Policy* définit trois opérations pour le traitement des politiques : *Normalize*, *Merge* et *Intersect*. Ces opérations définissent un modèle pour compiler les politiques en normalisant d'abord chaque politique et ensuite les combiner selon un ensemble de règles.

4. Le langage GXLA

4.1 *Le modèle d'information GSLA*

Le *GSLA* (Generalized Service Level Agreement) (Aib I et al., 2004) est un modèle d'information qui permet la spécification formelle des SLA et se prête à la traduction simple vers des règles de configuration de bas niveau. En se basant sur l'architecture PBM, il fait la liaison entre les besoins de la QoS niveau applicatif (Business) et la QoS offerte niveau réseau (ou liaison).

Le *GSLA* représente le contrat conclu entre deux ou plusieurs parties et définit un ensemble de mesures et de rôles clairs et compris par toutes les parties. Le rôle de chaque partie est un ensemble de règles qui définissent le niveau minimum du niveau de service attendu et le niveau de service nécessaire (obligatoire) à l'encontre des autres rôles sous certaines contraintes. Les contraintes peuvent être de n'importe quelle nature et normalement contiennent la portée du contrat (temporelle, géographique, etc.), l'accord sur les politiques de facturation et également le comportement du système en cas de défaillance. Figure1 (Aib I et al., 2004) présente les compo-

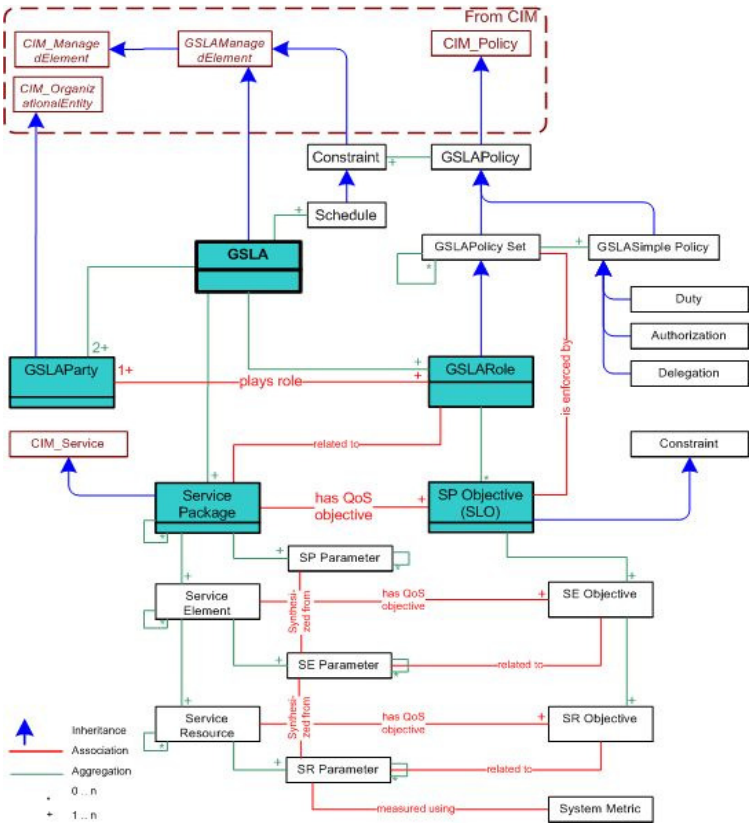


Figure1. *Modèle d'information GSLA*

sants de base du *GSLA*.

Le *GSLAParty* permet de décrire les parties (Fournisseurs, Client, etc.) impliquées dans le service. Contrairement aux autres spécifications vues précédemment, le *GSLA* ne définit pas une partie Client et une autre fournisseur mais il représente une modélisation de niveau d'abstraction plus élevée.

Le *GSLARole* contient un ou plusieurs *GSLAPolicySet*. Le rôle de chaque partie est défini par un ensemble de règles qui contrôle le comportement du système et hérite de l'ensemble *GSLAPolicySet*, les contraintes, comme la durée (*schedule*) de validité de la politique. Le GSLA modélise la garantie du niveau de service en associant à chaque service (*ServicePackage*) un *SPObjective* qui doit être assuré par certaines parties des signataires attachées au rôle.

Le *ServicePackage* est une abstraction qui permet au client de choisir un ou plusieurs services. Le *ServicePackage* regroupe un ou plusieurs *ServiceElement* en relation, qu'on pourra ensuite, instancier et gérer comme un ensemble ou offrir comme un tout aux utilisateurs, eg. Service Web, Service de messagerie et le service d'hébergement. Un *ServiceElement* peut être composé d'autres *ServiceElement* et, au niveau le plus bas, il sera relié à un ou plusieurs *ServiceResources*. Un *ServiceResource* est généralement transparent aux clients.

Les métriques (paramètres de performance) associées à un *ServiceElement* sont calculées à partir des métriques de ses constituants. Ainsi, des métriques du système sont combinées de façon hiérarchique afin d'évaluer les paramètres de performance d'un *ServiceElement*. Un *ServiceResource* est transparent aux utilisateurs et représente les capacités réelles du fournisseur (nombre de serveurs, nombre de routeurs, bases de données…). Cela permettra aux fournisseurs de conclure des contrats SLA et ensuite ajuster ses performances (SLS intrinsèques) en fonction de ses objectifs tout en respectant ses garanties. A chaque service correspond une qualité attendue par le client et promise par le fournisseur. Le GSLA modélise la qualité de service dans un ensemble de Service Level Objectives (SLO). La spécification des SLO est toujours confrontée au compromis entre la QoS attendue par le client, qui est en général formulée avec des paramètres de QoS de haut niveau, et la QoS côté fournisseur, qui est généralement décrite en fonction des performances bas niveau de ses équipements. A travers les SLO, le *GSLA* tente d'élever la spécification des performances du système jusqu'au niveau Business pour que les deux parties client et fournisseur puissent parler le même langage.

Dans le *GSLA*, la multi-partie est possible grâce aux SLO qu'assurent chaque partie et le comportement à suivre par rapport aux autres parties. En effet, à l'intérieur de chaque SLO existe un ensemble de règles (ou politiques) qui définissent l'action exacte à tenir en cas de non respect du SLO, ou certains seuils sont atteints. On regroupe les SLO en relation et leurs politiques dans l'élément *GSLARole*. Durant le cycle de vie du contrat, une partie peut jouer un ou plusieurs rôles. Cette définition du rôle dans le GSLA, indépendamment de la partie qui le prend en

charge, facilite la réaffectation du rôle en cas de besoin (eg. Déléguer les mesures à une partie tierce par exemple). Etant donné que chaque partie remplit un rôle dans le *GSLA*, ce rôle prend en compte toutes les possibilités de comportement que peut avoir une partie. Le comportement de chaque partie est modélisé dans le GSLA grâce aux politiques. Les politiques sont de deux types : Obligatoire (*Duty*) et Autorisation (*Authorization*). Conceptuellement, une politique d'autorisation met des conditions pour limiter l'accès ou l'exécution d'une action sur un élément du système. Les politiques d'*Authorization* peuvent être une permission ou une interdiction. Il est possible d'avoir plus d'informations sur le GSLA dans (Aib I et al., 2004).

4.2 *Modélisation XML du GSLA : le GXLA*

Le GXLA est un schéma XML qui implémente le modèle d'information GSLA. Il est utilisable à la fois par le (ou les) fournisseur(s) de service(s) et le(s) client(s) du service. En plus, il permet l'usage interne par le fournisseur de service pour la compilation et la configuration de son système afin de fournir les services conclus.

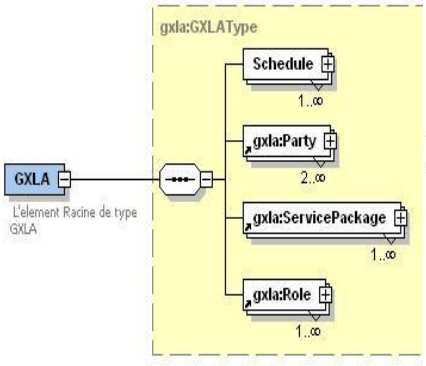


Figure2. *Définition globale du GXLA*

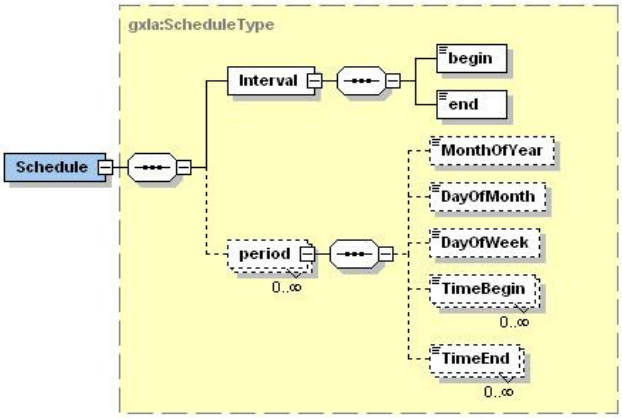


Figure3. *Schedule*

Le SLA dans le *GXLA* se compose de quatre sections (Figure2) :

1. *Schedule* : représente la portée temporelle globale du contrat. Elle est structurée comme un ensemble de points dans le temps délimités par un début et une fin. Cet intervalle est global qu'on peut ensuite raffiner en précisant le (s) mois (semaines, jours…) où le service est actif (valide). Le Schedule pourra ensuite être dérivé pour une portée restreinte concernant un service ou une contrainte (Figure3).

2. *GXLAParty* : représente les parties impliquées dans le service. Contrairement aux spécifications précédentes, le *GXLA* décrit les informations générales de chaque

partie sans préciser sa nature (fournisseur, client) pour couvrir tous les états possibles dans le marché des services réseaux. Il permet ensuite de rattacher chaque partie à un ou plusieurs SLO qui définissent sa nature.

3. *ServicePackage* : est une abstraction pour décrire un ou plusieurs services (Figure4). Il est composé de :

- *Constraint* : précise la portée temporelle de chaque service dans la limite de la portée temporelle des composants englobants, en l'occurrence la contrainte Schedule du SLA.

- *SP_Parameter* : est la formalisation des paramètres de haut niveau et représente la vue commune du service par tous les participants. Nous les définissons dans le *GXLA* comme des résultats de synthèse des métriques (*SE_Parameter*). Un *SP_Parameter* ou *SE_Parameter* peut être rattaché à un ou plusieurs SLO qui surveillent la QoS du service.

- *ServiceElement* : est une abstraction pour définir le service en question (eg. Hébergement, VoIP,…), ses opérations, ses contraintes et ses paramètres. Le *GXLA* permet de décrire le service avec des paramètres de haut niveau (vocabulaire du contrat) visibles aux clients. Ensuite, il rattache chaque service à une ou plusieurs ressources système (*ServiceResource*) d'une façon transparente aux clients. En effet, chaque paramètre du service (*SE_Parameter*) est le résultat d'une fonction ou d'une directive sur les métriques du système (eg. rapport SNMP, taille du flux sortant d'un routeur…). La métrique peut être atomique (eg. mesure sur une ressource) ou composée de plusieurs autres métriques (eg. moyenne du nombre d'invocations du serveur).

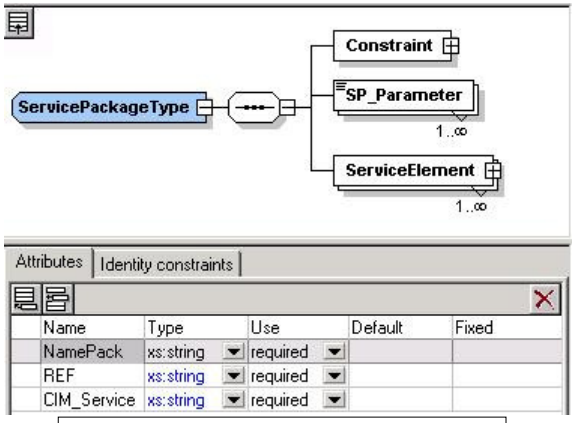


Figure4. *GXLAServicePackage*

4. *Role* : Pour chaque service, correspond une QoS garantie par le fournisseur et attendue par le client. Le *GXLA* décrit, dans la partie *Role,* les SLO qui vont définir

le niveau de service requis pour chaque service (Figure5). Un SLO, dans le *GXLA,* est assuré par une ou plusieurs politiques (Policy) (Figure6). Les politiques sont de plusieurs types : (i) Des politiques d'obligation (garantissent la QoS) qui définissent e comportement (*QualifedAction*) de la partie responsable du rôle sous certaines conditions sur les *SP_Parameter* (eg. exécution d'un schell, notification dans le cas d'une violation ou d'une valeur proche d'un seuil.). (ii) Des politiques d'autorisation qui définissent les droits d'accès et/ou d'actions sur une ressource système sous certaines conditions, en particulier sur les métriques (eg. changer la taille d'une fille d'attente).

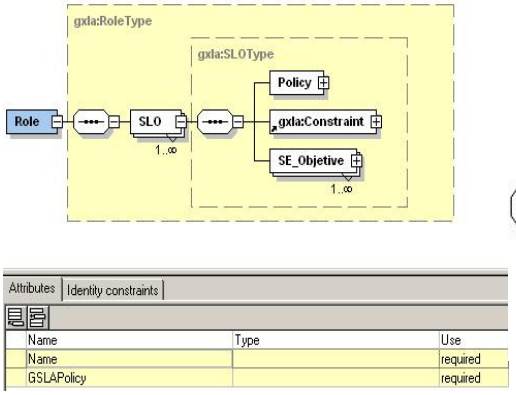


Figure5. *GXLA Role*

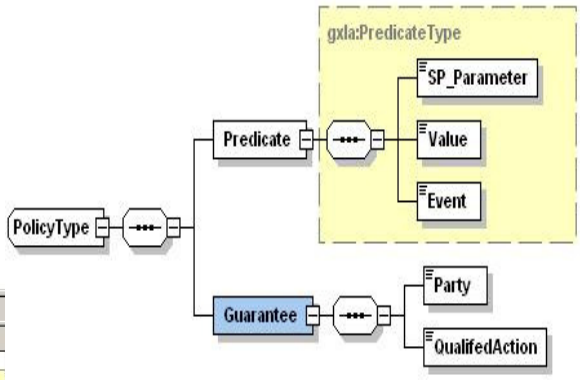


Figure6. *Policies*

5. Étude de cas

Nous considérons un *GXLA* entre un fournisseur de service et un client où est offert un service de VoIP.

Le client négocie sa demande de service à partir d'une interface graphique (servlet ou applet web) qui facilite la présentation de l'ensemble de modèles de SLA proposés par le fournisseur de service. Le client précise ses coordonnées, le service qu'il désir, et le niveau de QoS selon les paramètres disponibles (qualité, prix) et envoie sa demande. Le fournisseur reçoit la demande du client sous forme d'un script GXLA résumant le contrat choisi par le client parmi l'intervalle de choix qui lui a été offert. Ensuite, il renvoi au client la version du contrat GXLA à 'signer'. Si le client accepte le contrat est mis en action et le fournisseur de service procède à sa mise en exécution selon le calendrier (schedule) conclu dans le contrat.

Le fournisseur du service VoIP assure deux paramètres: Délai (*Delay*) et Niveau du Bruit (*Noise Level*). (Figure7) Représente la spécification complète du *GXLA* correspondant. Le Délai est mesuré grâce à la métrique *RTT*. Le paramètre de bruit est obtenu grâce à la métrique *PercentageNoise* qui est relevé chaque cinq minutes de l'url: *http://www.provider.org/readNpose?slaID=slaid&service=VoIP*. Chaque paramètre est relié à un SLO.

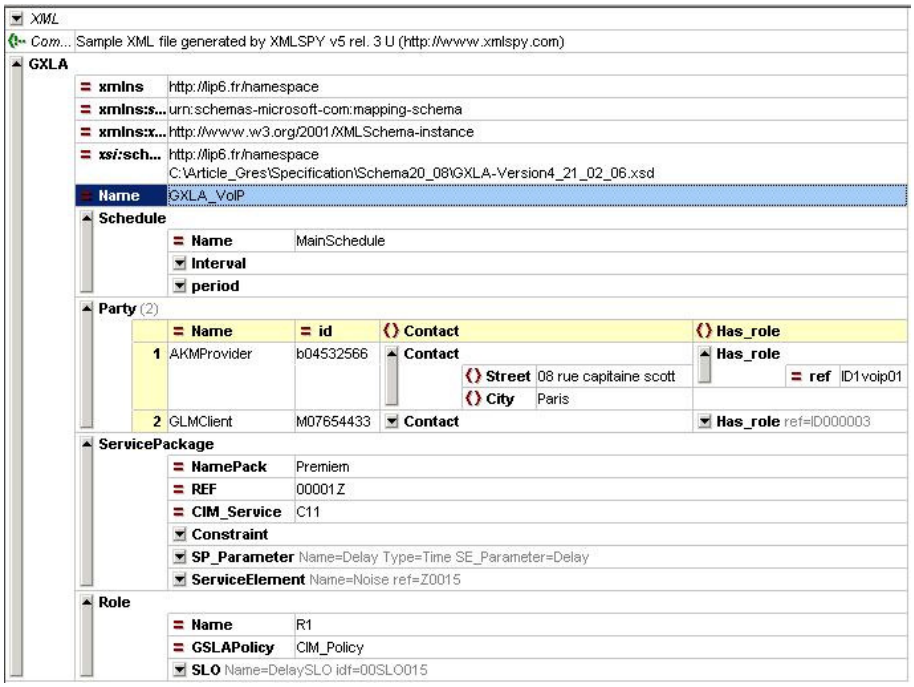


Figure7. *Spécification GXLA du cas d'étude. (Vue globale)*

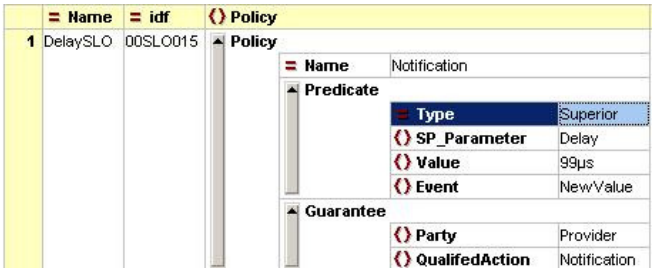


Figure8. *Spécification des SLO*

Le responsable est le fournisseur, il assure deux SLO (Figure8). Le premier concerne le paramètre Delay qui ne doit pas être supérieur à 128µs sur moyenne de 1 mois, avec un seuil d'alerte de 126µs. Dans notre spécification, l'action déclanchée dans le cas de violation du SLO est de notifier le fournisseur. Mais les actions peuvent être aussi bien intrinsèques au fournisseur comme l'exécution d'un script de configuration en cas de violation (ou surcharge) ou bien indépendante comme l'acceptation de nouveaux clients. C'est la même chose pour le deuxième paramètre Noise.

6. Conclusion et perspectives

Dans ce travail, nous avons présenté le schéma XML *GXLA* qui implémente le modèle d'information *GSLA* pour la spécification des contrats de niveau de service (SLA). Cette modélisation se base sur le concept de rôle. Chaque rôle englobe un ensemble de SLO et de règles qui caractérisent son comportement dans le SLA. Nous avons décrit l'usage du GXLA à travers un scénario d'utilisation dans la négociation d'un service *VoIP*.

La spécification du langage *GXLA* est la première étape dans le processus d'automatisation de la gestion orientée services. Nous considérons en perspective le développement d'une plateforme plus élaborée qui incluent en plus de la spécification des SLAs : leur raffinement en des politiques de gestion et de contrôle, leur optimisation, et enfin leur déploiement.

Références

Aib I., Agoulmine N. and Pujolle G., "The Generalized Service Level Agreement Model and its Application to the SLA Driven Management of Wireless Environments", ACIT, 2004.

Aib I., Agoulmine N. and Pujolle G., "Capturing Adaptive B2B Service Relationships Management through a Generalized SLA Information Model", HPOVUA, 2004.

Aib I., Agoulmine N. and Pujolle G., "A Multi-Party Approach to SLA Modeling Application to WLANs", IEEE Consumer Communications and Networking Conference, Jan 2005.

Bajaj S., and All., "Web Services Policy Framework (WSPOLICY)", september 2004, url = "citeseer.ist.psu.edu/703689.html.

Boutaba R., and Polyrakis A., "Extending COPS-PR with Meta-policies for Scalable Management of IP Networks", International Journal on Networks and Systems Management (special issue on Management of Converged Networks), Vol.10, No.1, pp. 91-106, 2002.

Dinesh C. Verma., "Simplifing Network Administration using Policy-Based Management", IEEE Network, march 2002.

Dan Richard A., Heiko L., Keller A., “A Service Level Agreement Language for Dynamic Electronic Services”, IEEE Network, 2002.

Distributed Management Task Force, Inc., “Common Information Model (CIM), Policy Model”, white paper, 18 June 2003. CIM Version 2.7.

Dulay N., Damianou N., Lupu E., and Sloman M., “A Policy Language for the Management of Distributed Agents”, In AOSE, pages 84-100, 2001.

GanzA., Ganz Z., and Wongthavarawat K., “Multimedia Wireless Networks: Technologies, Standards and QoS”. Prentice Hall PTR, 2004.

Heiko L., Keller A., Dan A., Dan Richard A., and Franck R.,” Web Service Level Agreement (WSLA) Language Specification”, IBM Corporation, 2003.

Internet Draft., “draft-ietf-policy-req-01.txt, Requirements for a Policy Management System”, Expire April 2000.

Internet Draft., Network Working Group., “draft-tequila-sls-03.txt, attributes of a Service Level Specification (SLS) template”, Expire April 2004.

James Skene D., Lamanna D., and Emmerich W., “SLAang : A Langage for Defining Service Level Agreements”, IEEE FTDS, 2003.

Lupu E., Sloman M., “The Ponder Policy Specification Language”, Departement of computing Imperial College 180 Queen's Gate London SW7 BZ Nicodemos, Naranker Dulay.. Proc. Policy 2001: Workshop on Policies for Distributed Systems and Networks, Bristol, UK, 29-31 Jan.2001, Springer-Verlag LNCS 1995, pp. 18-39, 2001.

MachirajuV., Sahai A., and van Moorsel A., “Web Service Management Network: An Overlay Network for Federated Service Management”, HPL-2002-234.

Moses T., “eXtensible Access Control Markup Language (XACML)”, version 1.0, Technical report, OASIS, Feb. 2003.

NETCONF : http ://www.ops.ietf.org/netconf/.

RFC 3060 Network Working Group., “Policy Core Information Model (PCIM)”, February 2001. Version1 Specification.

TeleManagementFORUM., “SLA Management Handbook GB917”, juin 2001.

TeleManagementFORUM and Lencent Technologies Bell Labs Innovations., “Service Level Agreements and Bechmarking in the IP Word”, Conférence en ligne, Avril 2005.

Modèle d'information pour le déploiement dynamique de services

Gladys Diaz— Nadjib Achir— Ken Chen

L2TI - Institut Galilée – Université Paris 13
99 Av J-B Clément – 93430 Villetaneuse- France
{diaz, achir, chen}@l2ti.univ-paris13.fr

RÉSUMÉ. *Nous traitons dans cet article de la spécification et de la modélisation de différentes notions dans le contexte des réseaux autonomes, et en particulier le contrôle du déploiement dynamique des services. Cet article présente un modèle d'information pour représenter les différentes notions impliquées dans ce contexte. Notre modèle d'information est centré sur les notions de service, ressources et profil. Notre modèle nous permet également de traiter la gestion des ressources réseaux. Nous présentons dans cet article une description du modèle en utilisant UML. Nous présentons également un exemple de modélisation des services en employant notre approche.*

ABSTRACT. *We are interested in the specification and modelling of different notions in the context of autonomic networks, in special to the dynamic deployment of services. This paper presents an information model to treat with the notions involved in this context. We propose an object-oriented information model to represent in a first time the notion of service, and to treat also the management of networks resources. The information model is representing by using UML. We present in this paper an example to model services by using our approach.*

MOTS-CLÉS : *déploiement dynamique de services, modèle d'information, modélisation orienté objet, UML*

KEYWORDS: *dynamic services deployment, information model, object-oriented modelling, UML*

1. Introduction

Les réseaux autonomes constituent un domaine de recherche actuellement émergent. Le terme *réseau autonome* fait référence aux réseaux qui ont la capacité de s'auto-configurer, et s'auto-gérer en accord avec la perception et l'évolution de leur environnement. Un tel système exige une administration minimale, impliquant généralement une gestion au niveau politique. Pour mettre en application cette configuration d'une manière automatique, il est nécessaire d'avoir une connaissance permanente et totale de l'environnement, c.-à-d. du réseau, et d'appliquer les fonctions nécessaires pour lui permettre d'avoir une démarche dynamique. Ainsi, un réseau autonome implique un nouveau niveau de traitement : c'est le *niveau de la connaissance*. La connaissance sur les services et les ressources de l'environnement devient donc primordiale pour gérer le réseau d'une manière autonome.

La gestion de réseau a été une des problématiques les plus étudiées dans les dernières années (Lewis, 2000) (Chen *et al.*, 2000). Dans ce contexte, plusieurs services peuvent être automatisés : la configuration des équipements, la gestion de QoS, etc. Pour automatiser ces services et réduire la complexité d'information à traiter, il est essentiel de modéliser les réseaux en décrivant les divers composants impliqués ainsi que leurs interdépendances. Actuellement un nouveau point d'intérêt vient s'ajouter à cette problématique, c'est le *déploiement dynamique des services* (Hass *et al.*, 2003). Nous nous focaliserons dans cet article sur la spécification et la modélisation des services pour leur déploiement dynamique. La modélisation constitue une première étape pour représenter le niveau de la connaissance sur des environnements en réseau et pour leur gestion automatique.

L'automatisation et le déploiement dynamique de services constituent un des objectifs finaux des fournisseurs des services. Cette dynamicité dans le déploiement exige une connaissance absolue sur leur environnement de déploiement. Pour répondre à ce besoin, un des problèmes à traiter consiste à déterminer les principales méthodes nécessaires pour l'analyse des stratégies de déploiement. Ces outils permettront d'analyser l'efficacité des décisions des placements au niveau de l'infrastructure, aussi bien matérielle que logicielles. De ce fait quantifier les conséquences d'un déploiement sur les ressources réseau et sur l'obtention des services visés.

En dépit du travail substantiel qui a été réalisé dans le secteur de la gestion de réseau et de service, un accord commun et général sur la terminologie n'a pas encore été atteint. La première notion à traiter est celle du *service*. Actuellement, cette notion est traitée différemment selon le domaine d'intérêt. Dans les systèmes répartis, par exemple, les spécifications et les aspects à modéliser sont tournés de plus en plus vers une perspective orientée service. Nous présentons par la suite notre point de vue sur la définition du service.

Dans cet article, nous présentons une analyse sur les différentes notions traitées. À partir de cette analyse, nous proposons un modèle d'information orienté objet

pour représenter la notion du service, et pour traiter également la gestion des ressources réseaux. Le modèle d'information est représenté en employant des diagrammes UML.

Cet article est organisé comme suit. La section 2 présente le contexte de notre travail. Nous présentons dans cette section nos motivations, ainsi qu'un aperçu des recherches en cours dans ce sujet. La section 3 présente le modèle d'information proposé. Nous proposons les définitions pour les notions introduites par notre modèle. La section 4 présente un exemple illustrant l'utilisation de notre modèle. Finalement, la section 5 présente nos conclusions et perspectives.

2. Contexte du travail

Un *service dynamique* constitue une nouvelle notion « réseau » qui donne encore plus de contraintes à la gestion autonome des réseaux. Cette notion est utilisée actuellement dans différents contextes : web services, réseaux mobiles, et en général dans le contexte des réseaux autonomes. La définition même de *service dynamique* implique deux questions : d'abord quelle est la définition du service lui même? et en second lieu, qu'est ce qu'est la dynamicité et comment elle peut être mise en place ? Nous avons présenté dans (Diaz *et al.*, 2006) une étude pour essayer de répondre à ces deux questions.

La notion de *service* dénote, d'une manière générale, une fonctionnalité d'un système. Cette notion peut apparaître à différents niveaux (niveaux d'exécution, niveaux d'abstraction) et peut être considérée également dépendante du contexte d'application. Ainsi par exemple, un service de haut niveau (ex. VoIP) peut être réalisé de différentes manières, par plusieurs composants et protocoles. En modélisant les besoins d'un tel service sous forme de services élémentaires et de disponibilité des ressources, l'activation du service pourra être réalisée dans des contextes très hétérogènes, par exemple en sélectionnant l'option la mieux appropriée aux ressources et au contexte, en interconnectant des services répondant aux mêmes besoins, ou en activant des passerelles.

Le *déploiement dynamique* implique la possibilité pour un réseau d'assurer, à partir d'un processus automatisé, l'installation des services demandés par les clients. Ce processus implique la connaissance sur la sémantique du service, et sur les éléments du réseau (et/ou des autres composants impliqués) et les ressources nécessaires. Ainsi, pour réaliser un déploiement dynamique, il est nécessaire de faire une configuration de chaque élément impliqué dans ce déploiement. Cette configuration implique de prendre des décisions sur les éléments qui doivent être choisis, en fonction de la connaissance sur les ressources disponibles de ces éléments. Nous avons présenté dans (Diaz *et al.*, 2005) une étude sur les contraintes liées au déploiement dynamique de services.

Actuellement, le déploiement dynamique de services implique deux contextes de travail :

- Le déploiement de *composants logiciels* pour des applications réparties. Dans ce cas, une application est vue comme une composition de plusieurs « composants logiciel », qui doivent être assemblés. Le problème de base dans ce contexte *est comment faire cette composition pour assurer le service désiré ?* Ici la sémantique du service est créée et doit être vérifiée par des méthodes de composition. On trouve dans la littérature différentes propositions dans ce sens (Cervantes *et al.*, 2003), (Fokus, 2003), (Mennie *et al.*, 2000).

- Le déploiement dynamique des *services réseau*. Dans ce cas, la problématique est située dans des environnements réseau. Le problème principal dans ce contexte est *comment distribuer le service demandé* ? Et *quelle est sa meilleure localisation* ?, Afin de garantir leur fonctionnement en accord avec des contraintes spécifiées par le demandeur du service (Bennani *et al.*, 2000) (Hass *et al.*, 2003) (Kon *et al.*, 2005). Dans ce papier nous traitons le deuxième aspect.

En ce qui concerne les modèles d'information, plusieurs normes (OSI, ITU-TS, IETF) ont proposées des modèles pour la gestion réseau. Certains travaux s'intéressant au niveau équipement : (Aschemann *et al*, 1998) présente un modèle d'information orienté objet pour la gestion de configuration et pour décrire des contraintes liées aux ressources dans des environnements distribués. Des autres modèles, considérant non seulement le réseau mais également des niveaux plus élevés : (Gbaguidi *et al.*, 1996) propose un modèle de l'information pour le contrôle des services de télécommunication. Ce modèle, orienté QoS, permet d'indiquer tant les exigences d'utilisateurs que des contraintes liées aux ressources réseau. (Maknavicius *et al.*, 1998) propose une séparation des niveaux de gestion, avec un modèle qui permet de définir l'information à chaque niveau : gestion de session d'utilisateur, de la communication et du réseau.

Actuellement, des travaux tels que (Garschhammer *et al.*, 2001), présentent quelques définitions commune avec notre modèle. Garschhammer propose un modèle générique de gestion de service dans les environnements des télécommunications. Ce travail décrit bien les contraintes côté client et fournisseur de services, mais peu de détails sont donnés au sujet du déploiement dynamique des services et des contraintes liées au réseau.

La motivation principale de notre travail est de pouvoir exprimer, avec un modèle général, l'information nécessaire à la configuration des services distribués/déployés. Afin de traiter cette problématique, nous avons défini un modèle d'information permettant de caractériser l'ensemble du réseau et de ses composants matériels et logiciels. Ce modèle s'articule autour des notions de *profil de service*, *profil d'équipement* et *profil de déploiement.* L'idée principale est de pouvoir prendre en compte l'ensemble des ressources disponibles et leur évolution lors de l'activation de services distribués. Les principaux apports du modèle d'information ainsi définie sont :

- d'être un modèle général, dont la notion de service est vue séparément de l'instanciation du service ;

- de prendre en compte la notion de service composé (via les notions d'héritage et de services élémentaires) ;

- de prendre en compte des ressources multiples lors du déploiement ;

- de combiner des informations de localisation avec des informations sur les ressources disponibles/consommées.

3. Modèle de service

3.1. *Description du modèle*

Nous présentons dans cette section notre proposition de modèle d'information (*ProfilModel*) pour représenter les notions de profils de service, de déploiement et d'équipement. L'objectif principal du modèle est de représenter d'une part l'état actuel de l'environnement du réseau (du point de vue des services existants et des ressources consommées et disponibles), et d'autre part, les informations nécessaires à la mise en place de nouveaux services, ou des nouveaux profils pour des services déjà existants.

Nous définissons dans les sous-sections suivantes les hypothèses prises en compte pour la définition du modèle et la terminologie utilisée, qui correspondent à l'ensemble de classes définies par notre modèle. Nous commençons dans la suite la description de notre modèle par la définition d'un diagramme de classes, voir figure 1. Ce diagramme va nous permettre de collecter le vocabulaire du système, de modéliser leurs concepts et de spécifier leurs schémas logiques. À travers ce diagramme, nous présentons la vue logique et déclarative des définitions des objets qui seront utilisés par la suite.

3.2 *Hypothèse sur le modèle*

Pour la spécification de notre modèle certaines hypothèses on été considérées :

- Tous d'abord, nous considérons dans ce travail une gestion centralisée du réseau. De ce fait, un équipement du réseau central (gestionnaire) est chargée de stocker la configuration actuelle du réseau sous la forme d'une base de données représentant le modèle d'information introduit précédemment. L'instanciation de cette base de donnée peut se faire à travers l'ajout d'un mécanisme de découverte de service. De plus, tout changement dans la configuration du réseau (i.e. introduction d'un nouveau service) doit être soumis dans un premier temps au gestionnaire du réseau. Ce gestionnaire peut jouer le rôle d'un PDP (Policy Decision Point). Dans le cas d'une architecture PBM.

- Nous considérons également une connaissance sur les relations entre un service composé et les services simples qui le composent. Notre modèle permet donc de décrire les relations de composition de services.

- Finalement, nous considérons l'existence de services de découverte et de déploiement. Plus exactement, nous supposons l'existence dans le réseau de mécanismes capables d'interagir et d'enrichir le modèle d'information avec les différentes informations décrites dans notre modèle. La notion de profil proposé par notre modèle facilite donc les tâches de ces mécanismes en indiquant quels paramètres doivent être mis à jour.

3.3. *Terminologie*

3.2.1. *Service*

Nous proposons, dans notre approche, une définition générale de service. Un service est défini comme étant tout code réalisant une fonctionnalité exploitable, sans pour autant tenir compte du niveau d'implémentation. Il est constitué de plusieurs éléments : (a) un code, qui décrit les algorithmes d'implémentation du service ; (b) un état. En effet, un service peut se trouver dans des états différents tout au long de sa vie (runnable, stopped, sleeping, canceled).

De plus, un service est caractérisé par plusieurs propriétés, quelque unes statiques (partie constante) et quelques autres dynamiques (liées et variantes selon son instanciation) : (a) partie constante : son Identificateur, son API, organisme créateur, version du service ; (b) partie dynamique : décrite à travers la notion de *profil de service*. Cette partie est liée à l'environnement où évolue le service en question. Par exemple, la quantité de ressources utilisées par ce service, sa localisation.

Cette définition de service est représentée dans la **figure 1**, par la classe *ServiceType*, qui désigne à la fois un service simple ou un service composé, et leurs interrelations.

3.2.2 *Profil*

D'une façon plus générale, un profil permet de décrire une entité (matériel ou logiciel) en définissant un modèle adapté à un certain type de besoin. Ainsi, un même service peut avoir plusieurs profils, dépendant des besoins et contraintes spécifiés. Et un profil donnera lieu à une configuration particulière pour l'implémentation d'un service. Dans le cadre de notre étude nous allons travailler avec trois types de profils : *Profil de service, Profil d'équipement*, et *Profil de déploiement*. Nous présentons dans la section 4 des exemples pour ces définitions.

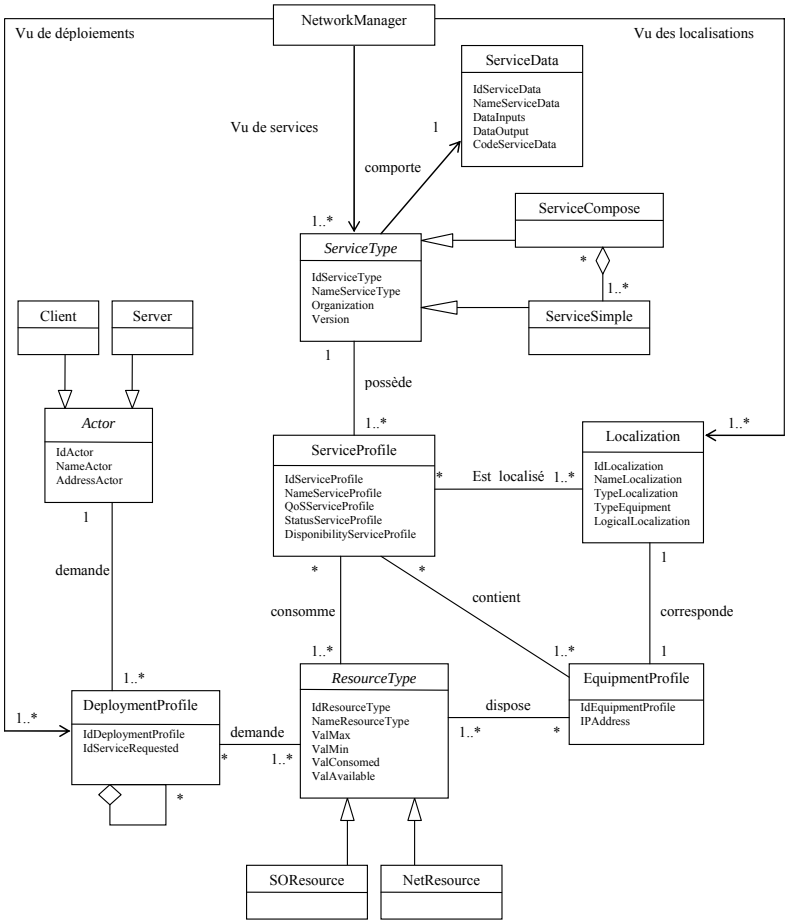


Figure 1. *Modèle d'information – Le diagramme de classes*

3.2.2.1 Profil de Service

Un profil de service permet de décrire une configuration particulière pour l'exécution d'un service. Chaque configuration spécifie les contraintes et besoins liées à l'exécution du service. Un profil de service sera caractérisé par : l'ensemble des ressources sollicitées pour l'exécution du service, une localisation (physique et logique), la QoS associée, des contraintes sur certaines paramètres (fournisseur du service, couverture géographique, prix du service, disponibilité / durée du service). Cette définition est représentée dans la **figure 1**, par la classe *ServiceProfile.*

3.2.2.2 Profil d'équipement

Un profil d'équipement permet de définir la configuration d'une entité matérielle (équipement). Cette configuration englobe plusieurs aspects : les composants

physiques (CPU, Stockage, Batterie, Affichage, etc.), les composants logiciels (correspondant à l'environnement qui est vu à travers la liste de profils de services offerts dans cet équipement). Cette définition est représentée dans la **figure 1**, par la classe *EquipmentProfile.*

3.2.2.3 Profil de déploiement

Un profil de déploiement permet de capturer les requêtes des services. Ce profil permet de décrire la requête en termes de ressources et de services demandés. Ainsi nous pouvons décrire les besoins d'un nouveau service devant être déployé en termes de ressources (software/hardware), et d'autres services, nécessaires à leur instanciation dans le réseau. Ce profil est fourni par le client au gestionnaire du réseau, et est représenté par la classe *DeploymentProfile* dans la **figure 1.** Il est à noter qu'un profil de déploiement peut être composé d'autres profils de déploiement dans le cas où le service demandé est un service composé.

4. Exemple de modélisation

4.1 *Contexte d'étude : Le serveur vidéo*

L'avènement du multimédia, conjugué à la démocratisation rapide du haut débit dans l'Internet, reste certainement l'un des événements les plus marquants durant la dernière décennie dans le domaine de l'informatique. Un grand nombre d'applications multimédias, en particulier les applications vidéo, ont ainsi vu le jour comme la vidéo à la demande (Video-On-Demand, VOD), la gestion d'archives audiovisuelles, la vidéo surveillance, etc.

4.1 .1 *Contraintes à modéliser*

Au coeur de tout système vidéo, le *Serveur Vidéo* constitue la partie centrale responsable du stockage, de la gestion, et la diffusion des données audiovisuelles.

Les données audiovisuelles ont une nature intrinsèquement différente de celles des données dites conventionnelles (texte, image, etc.), leur gestion impose souvent le développement de techniques spécifiques. Il est difficile d'imposer une version *catalogue* des serveurs vidéo tant ceux-ci sont à géométrie variable. La RAM, la mémoire de masse, l'architecture de stockage des données, le format de compression sont autant de paramètres qui changent à la demande selon les contraintes de l'application. On peut néanmoins en citer les principaux composants techniques :

1. La puissance du processeur : Paramètres => X Ghz.
2. La quantité de mémoire vive RAM ou la mémoire tampon : Paramètres => Mbytes.

3. La caractéristique des données audiovisuelles : codeur-décodeur (par exemple MPEG-2, MPEG-4 ou H264).

4. Les unités de stockage : Paramètres =>X Gbytes. Alors que la puissance des processeurs double tous les ans, l'unité de stockage limite encore fortement les performances du serveur. Plusieurs supports sont en concurrence pour le stockage de la vidéo numérique : les disques durs, la cartouche de données, la bande-vidéo, et, récemment, le disque optique avec le DVD.

5. Transmissions sur le réseau de distribution : les réseaux actuels sont souvent optimisés pour le trafic de données fragmentables et non pour le trafic de flux continus. Lorsqu'il y a saturation du réseau, les flux transportant des données temps réel ne peuvent plus êtres délivrés en continu. Depuis la forte évolution du multimédia, de nombreux travaux ont été faits dans le cadre notamment des groupes de travail de l'IETF pour adapter les réseaux aux besoins des applications multimédias. Par exemple :

a. Transmissions multipoint (multicast) : Paramètres => Existence d'un protocole de multicast. Dans un réseau de distribution, l'objectif est de réduire l'utilisation de la bande passante par chaque client. De ce fait un protocole de diffusion multipoint est primordial pour un serveur vidéo. Les algorithmes de routage multipoint actuellement utilisés (PIM, PIMSSM, MOSPF), sont basés sur l'adressage IP multipoint.

b. *Protocoles spécifiques d'adaptation :* Paramètres => Existence du protocole RTP/RTCP. Face aux lacunes de protocoles tels que TCP adaptés ni aux données temps réel ni à l'adressage multipoint, les protocoles RTP/RTCP (Real Time Protocol/Real Time Control Protocol) ont été développés. Ces protocoles, indépendants du réseau et des couches de transport sous-jacent permettent une adaptation du transfert vidéo en fonction des ressources disponibles.

4.1 .2 *Modélisation*

À partir des composantes techniques précédemment introduites nous pouvons dégager un certain nombre de concepts liés au modèle. Nous montrons dans les figures suivantes quelques exemples d'objets définis pour cet exemple en utilisant notre modèle.

Tout d'abord, tel que nous avons décrit le service *serveur vidéo*, nous constatons que celui-ci est sous la forme d'un service *composé*. Plusieurs services simples sont requis pour l'instanciation d'un serveur vidéo. La **figure 2** montre la relation entre le service composé *serveur vidéo* (appelé SV) et les services simples qui le composent : type de codeur-décodeur (TV-MPEG-2 ou TV-MPEG-4), le service d'adaptation (SA) qui définie quel(s) protocole(s) peut être utilisés pour le transfert de la vidéo, et le service multicast (SM).

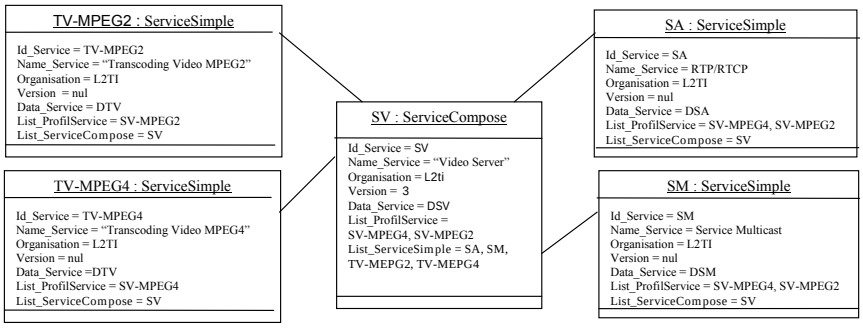


Figure 2. *Relation entre le service composé Serveur Vidéo et les services simples SA, SM et TV (MPEG2 et MPEG4).*

La **figure 3** montre un exemple de définition de profil pour le *service simple TV-MPEG-4*, le profil est appelé SV-MPEG-4. Chaque profil fait référence à l'ensemble des ressources consommées, dans notre exemple, ces ressources correspondent à la puissance du processeur (Processeur-1), à la taille de la mémoire RAM (RAM-1) et à la taille de stockage en disque (Disk-1).

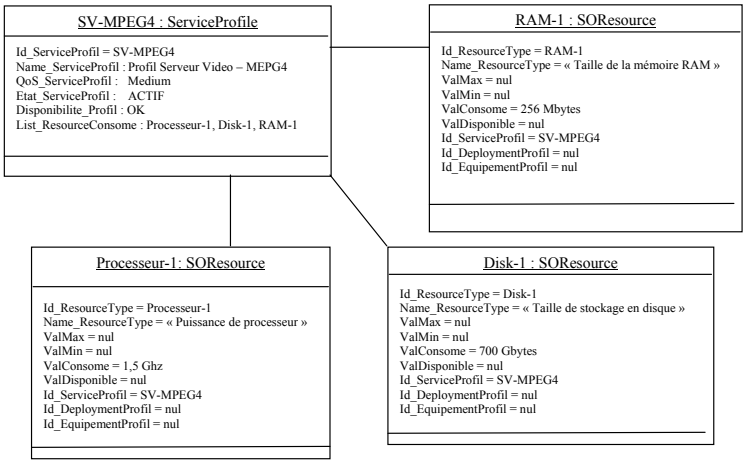


Figure 3. *Définition du service profil MPEG-4.*

La **figure 4** montre un exemple de définition pour les notions de profil d'équipement et leur localisation. Dans cet exemple nous décrivons les relations entre le profil de service SV-MPEG4, sa localisation physique (appelé PC-VoD 02) et finalement le profil d'équipement défini pour cette localisation (appelé PC Profil MPEG4). Le profil d'équipement nous permet de représenter la liste des ressources disponibles pour la localisation correspondante. Nous montrons, dans cette figure, deux exemples de ressources : Taille de mémoire (RAM-5) et Taille de stockage en disque (Disk-5).

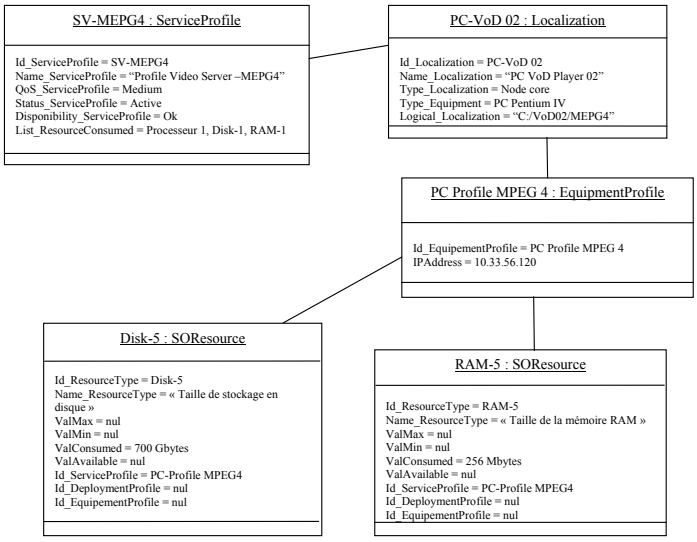


Figure 4. *Relation entre le profil de service « MPEG-4 », sa localisation et le profil d'équipement correspondant.*

5. Conclusions et perspectifs

Nous avons présenté dans cet article notre modèle d'information. Notre motivation principale est de pouvoir faciliter le déploiement des services réseaux dynamiquement. Ce déploiement dynamique de service est basé sur la caractérisation des besoins de service et l'état du réseau. Dans ce modèle, nous proposons le concept de *profil de service, profil d'équipement,* et de *profil de déploiement*. Les profils de service identifient une configuration possible du service (quantité de ressources, de paramètres de configuration, etc..). Les profils de déploiement identifient une demande de service d'un client. Ainsi, en utilisant cette information notre modèle fournit une vision *en temps réel* sur l'état du réseau et sur l'acceptation ou pas des nouveaux services dans le réseau.

Cet outil permet d'établir, pour toute demande de nouveau service, l'ensemble de composants nécessaires avec leur état (localisation, disponibilité de ressources, etc.), ainsi que leur relation d'interdépendance. Cette cartographie permettra à un gestionnaire de prendre des décisions de déploiement efficaces.

Nous travaillons actuellement sur une implémentation du modèle d'information. Cette implémentation est basée sur une approche centralisée du contrôle des ressources et services. Nous avons également entamé une étude sur l'adaptation de notre modèle pour introduire des nouvelles contraintes. Pour la suite de ces travaux, plusieurs points se présentent comme des perspectives intéressantes : (a) La mise en place d'une approche distribuée. Nous avons entamé cette perspective en étudant

des approches peer-to-peer et en adaptant notre modèle en conséquence. (b) La définition plus précise des paramètres de QoS au niveau des profils de déploiement et de service. (c) L'adaptation de modèle pour représenter la connectivité entre les nœuds du réseau, et pouvoir représenter les relations entre les profils d'équipement.

6. Bibliographie

Aschemann G.,, Kehr, R. "Towards a Requirements-based Information Model for Configuration Management". Proceedings of 4th International Conference on Configurable Distributed Systems, IEEE Computer Society Press, May 1998, 181-189.

Bennani F., Simoni N. "Dynamic management for end to end IP QoS : from best-effort to personalized services", In Terena Networking Conference 2000, Lisbon, 22-25 May 2000.

Cervantes H., Hall R.S. "Automating Service Dependency Management in a Service-Oriented Component Model", ICSE CBSE6 Workshop 2003, Portland, USA.

Chen G., Kong Q. "Integrated Management Solution Architecture", IEEE/IFIP - NOMS 2000, pages 217–230, Honolulu, Hawaii, USA, April, 10–14 2000. IEEE.

Diaz G., Achir N., Chen K. "Déploiement contrôlé des services de réseau", Livrable L1.1 du projet RNRT/Amarillo, mars 2005.

Diaz G., Achir N., Chen K. "Modeling Data to Management Dynamic Services Deployment in Autonomic Networks", à apparaître dans IEEE/ICTTA 2006, 24-28 april 2006.

Fokus F. "Deployment & configuration of component-based distributed applications specifications", OMG specifications, ptc/03-07-08, july 2003.

Garschhammer M., Hauck R., Hegering H.-G., Kempter B., Langer M., Nerb M. Radisic I., Roelle H. and Schmidt H. "Towards generic Service Management Concepts – A Service Model Based Approach". Proceedings of the 7th International IFIP/IEEE Symposium on Integrated Management (IM 2001), Seattle, Washington, USA, May 2001.

Gbaguidi, C., Znaty S., Hubaux J-P. "Service Management: From Definition to Information Modelling". IEEE GLOBECOM '96 Conference, London, Nov. 1996.

Hass R., Droz P., Stiller B. "Autonomic Service Deployment in Networks". IBM Systems Journal, Volume 42, Issue 1 (January 2003) Pages: 150 – 164, ISSN:0018-8670.

Kon F.,Marques J-R. Yamane T., Campell R.H., Mickunas D. "Design, implementation, and performance of an automatic configuration service for distributed component systems", In Software: Practice and Experience, 35(7), pp. 667-703, May 2005.

Lewis D. "A Review of Approaches to Developing Service Management Systems". Journal of Network and Systems Management, 8(2):141–156, 2000.

Maknavicius, L. and Znaty, S. "Generic Information Architecture for Service Management". HPOVUA'98 Workshop, 19-21 April 1998, Rennes, France.

Mennie, D., Pagurek, B., "A Architecture to Support Dynamic Composition of Service Components", proceedings of the 5th International Workshop on Component -Oriented Programming (WCOP 2000), Sophia Antipolis, France, June 13, 2000.

Impact des pertes de paquets sur le comportement des utilisateurs

Denis Collange, Jean-Laurent Costeux

France Télécom R&D
905, rue Albert Einstein
F06921 SOPHIA ANTIPOLIS CEDEX
{denis.collange, jeanlaurent.costeux}@francetelecom.com

RÉSUMÉ

Nous montrons dans cet article l'influence des pertes de paquets sur le comportement des utilisateurs ADSL, et plus précisément sur les caractéristiques de trafic de leurs connexions TCP: durée, taille et inter-arrivée. Lorsque le taux de perte augmente, les connexions sont plus petites, durent moins longtemps, et sont plus espacées. La réduction des tailles n'est pas seulement due à l'interruption de connexions en cours mais aussi à une certaine forme d'autocensure. Par conséquent, le taux de perte est un critère de performance pertinent pour détecter des problèmes de performance dans le réseau. Divers seuils sur les pertes sont définis selon l'impact plus ou moins important du taux de pertes. Ces seuils pourront être utilisés pour la détection automatique des problèmes de performance. Nous montrons également que cet impact dépend de l'application.

ABSTRACT.

We show and analyze in this paper the influence of the losses on the behaviour of ADSL users and more precisely on the traffic characteristics of their TCP connections: duration, size and inter-arrival. When the rate of loss increases, connections are smaller, shorter, and are spaced. The reduction of the sizes is not only due to the abortion of connections but also to a certain form of self-censorship. The loss rate may then be used to detect performance problems on networks. Various thresholds are defined, depending on the impact of loss rate. These thresholds could be used for the automatic control of performance problems. We also show that this impact depends on the application

MOTS-CLÉS : détection des problèmes, pertes de paquet, trafic.

KEYWORDS: problem detection, loss rate, traffic.

1. Introduction

Il est difficile pour un opérateur de réseau de définir a priori un seuil sur un critère de performance assurant que le réseau fonctionne correctement, et les clients sont satisfaits, si et seulement si la performance mesurée est meilleure que ce seuil. En effet, les équipements du réseau sont utilisés à tout instant par un très grand nombre de clients, chacun avec ses propres besoins, applications, sites distants... Un client peut donc être perturbé par un certain niveau de performance qui satisfera tout à fait d'autres clients. L'intérêt pour l'opérateur de définir un seuil, éventuellement plusieurs pour différents niveaux de gravité, est d'automatiser la surveillance des performances de bout en bout. L'objectif est surtout de détecter les problèmes de performance, afin de permettre une gestion proactive du réseau : différentes actions peuvent ensuite être décidées par l'opérateur du réseau, ou de manière automatique, selon le seuil dépassé pour les corriger.

Une méthode possible pour définir de tels seuils peut être de supposer que les utilisateurs modifient leur comportement lorsqu'une dégradation des performances devient gênante. Ils peuvent abandonner des transferts en cours s'ils les trouvent trop longs, ou s'autocensurer, donc réduire la taille moyenne des transferts. Les utilisateurs peuvent ensuite renouveler des transferts interrompus, ou au contraire, attendre la fin de transferts en cours pour en générer de nouveaux. Naturellement, cette modification du comportement n'est pas seulement une indication de mauvaises performances du réseau, mais aussi une manifestation d'insatisfaction des clients. De plus, les transferts interrompus consomment inutilement des ressources, et perturbent donc les transferts aboutis. L'opérateur à donc tout intérêt à agir, tant pour satisfaire ses clients que pour optimiser l'usage du réseau. Les analyses de trafic précédentes considérant l'impact des performances sur le comportement des utilisateurs ont surtout considéré les interruptions de connexions (Rossi, 2002), (Peukheuri, 2002). D'autres auteurs ont analysé divers modèles de cet impact, mais sans se référer à des observations sur un réseau opérationnel (Guillemin et al., 2003), (Roberts et al., 2003) et (Yang et al., 2001). Nous avons observé (Collange et al. 2005) que pour un réseau grand public, le délai de bout en bout, bien que souvent proposé dans la littérature, n'est pas un critère pertinent pour détecter les problèmes de performance dans le réseau. En effet, une forte proportion des transferts sont soit dus à des applications pair-à-pair, soit parallèles à des transferts pair à pair de l'utilisateur. Par conséquent, quelle que soit la statistique considérée, le délai de bout en bout est élevé et varie peu.

Notre premier objectif est tout d'abord de montrer que les pertes de paquet ont bien un impact sur le comportement des utilisateurs, tout au moins sur la distribution des tailles et le taux d'arrivée de leurs transferts. Le second point sera de quantifier cette influence : définir des seuils à partir duquel elle est perceptible, et la sensibilité des clients et à une dégradation des performances du réseau, qui doit naturellement dépendre des applications utilisées. Le troisième point de notre étude sera donc d'étudier la sensibilité des principales applications.

2. Méthode de mesure

Nous décrivons tout d'abord le réseau arborescent de collecte du trafic ADSL, sur lequel sont effectuées nos captures de trafic. Nous présentons ensuite notre méthode générale de capture et d'analyse de trafic, puis nous précisons comment sont estimés les taux de pertes de paquets.

Le DSLAM (Digital Subscriber Line Access Multiplexer) est le premier équipement du réseau de collecte ADSL, multiplexant le trafic de plusieurs centaines de clients. Le trafic de plusieurs DSLAM est ensuite multiplexé vers un BAS (Broadband Access Server), gérant les accès des clients. Les BAS d'une plaque locale (grande ville) sont ensuite reliés au reste de l'Internet par un routeur. Nous avons capturé et analysé dans ce document tout le trafic d'un BAS de la plaque ADSL de Nice.

Les mesures analysées dans cet article sont issue de la plateforme DIP de capture, d'analyse et de stockage long-terme du trafic (plus précisément des en-têtes des paquets IP) que nous avons développée à partir de la librairie pcap (Jacobson et al., 1989). A partir de ces traces de trafic, un certain nombre d'indicateurs caractérisant le trafic et les performances des connexions TCP et UDP sont calculés au vol.

En particulier, pour estimer les taux de pertes par connexion, sur lesquels sont basées nos analyses, nous avons mesuré la proportion de paquets hors séquence par connexion. Un paquet TCP est "hors-séquence" lorsque son numéro de séquence TCP est inférieur à celui du paquet précédent. Cette mesure n'est donc possible que pour les transferts TCP, qui représentent tout de même 95% du trafic. Nous avons également effectué les analyses suivantes avec une autre méthode d'estimation des taux de pertes, basée sur les retransmissions, et nous avons obtenu des résultats similaires. Nous souhaitons évaluer l'influence du taux de pertes global sur les caractéristiques des connexions à un instant donné. Cependant, comme observé à la sous-section 3.1, ces caractéristiques varient assez peu au cours de la journée. Nous avons donc plutôt considéré les corrélations par connexion, entre ses caractéristiques de trafic et son taux de pertes.

3. Analyse toutes applications confondues

Comme observé par (Peukheuri, 2002), l'utilisateur est surtout sensible au temps de réponse, i.e. au délai pour terminer sa transaction. Nous avons donc tout d'abord considéré l'impact des pertes sur la durée des transferts. Cependant, comme nous allons l'observer, ce temps de réponse est lui-même influencé par les délais de retransmission des paquets perdus lorsque le taux de pertes augmente. C'est pourquoi nous avons dans un second temps considéré le comportement des utilisateurs à travers la taille des connexions TCP. Nous étudierons ensuite l'impact des pertes de paquets sur les interruptions de connexions, puis sur le taux d'arrivée.

3.1. Caractéristiques globales du trafic

La Figure 1 représente l'évolution sur 24 heures, et par rapport à la moyenne sur la journée, du taux de pertes moyen et des moyennes par heure des caractéristiques de trafic considérées par la suite: la durée, volumes montant et descendant des transferts, et inter-arrivée entre transferts consécutifs d'un client. La moyenne et le coefficient de variation sur la journée de ces grandeurs sont donnés dans le 0. Nous remarquons sur cette figure qu'elles varient assez peu, de plus ou moins 40% par rapport à la moyenne. La plus grande variation est celle de la durée moyenne des transferts qui double en fin de journée, sans corrélations apparentes avec les autres caractéristiques. Ceci laisse supposer une dégradation des débits des transferts, due à la montée en charge du trafic résidentiel le soir. Nous remarquons aussi sur cette figure que les transferts sont un peu plus gros la nuit que le jour.

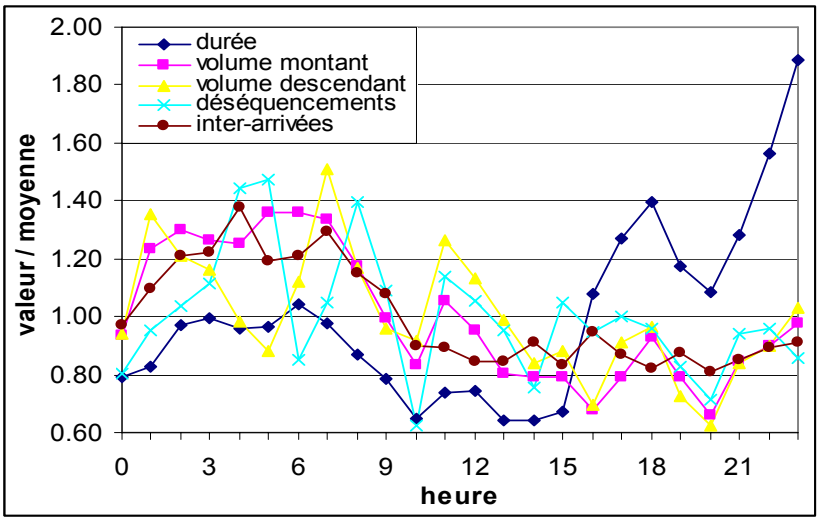


Figure 1. *Evolution des caractéristiques des connexions au cours de la journée*

tableau 1. Moyenne et coefficient de variation sur la journée des caractéristiques des transferts

heure	Durée (secondes)	Volume montant (kBytes)	Volume descendant (kBytes)	Taux de pertes	Inter-arrivée (secondes)
moyenne	73.2	9.45	16.2	6.30E-04	28.2
cv	0.31	0.23	0.21	0.21	0.17

3.2. Distribution des durées des connexions TCP

La Figure 2 représente la distribution des durées des connexions, en secondes, pour un taux de pertes donné, indiqué en légende. Nous observons que, tant que le taux de pertes reste inférieur à 10^{-3}, les connexions durent pour la plupart plus de 1000 secondes. Pour des taux de pertes supérieurs, les courbes sont progressivement décalées vers la gauche, ce qui indique que les connexions sont de plus en plus courtes. Nous constatons que pour des taux de pertes supérieurs à 10^{-2}, les courbes restent proches, avec des durées de connexions comprises pour la plupart entre 30 et 100 secondes. Nous en déduisons une nette corrélation entre la durée et le taux de pertes d'une connexion. Cependant, rien sur ces courbes n'indique que des taux de pertes élevés ont effectivement conduit à interrompre des connexions en cours. En effet plusieurs raisons peuvent conduire aux mêmes observations. Tout d'abord, les utilisateurs, conscients de mauvaises performances antérieures peuvent s'autocensurer en générant préférentiellement de petits transferts. D'autres explications sont plus inhérentes aux applications elles-mêmes. En effet, certaines applications, et notamment celles qui génèrent le plus de trafic (les applications pair-à-pair), régulent elles-mêmes leur débit. Ceci peut leur permettre d'obtenir des taux de pertes plus faibles que, par exemple, de courtes requêtes web souvent lancées en parallèle pour charger simultanément différents éléments d'une page HTML. Nous considèrerons l'impact des pertes sur l'interruption des connexions à la sous-section 3.4, et l'impact différencié selon l'application à la section 4.

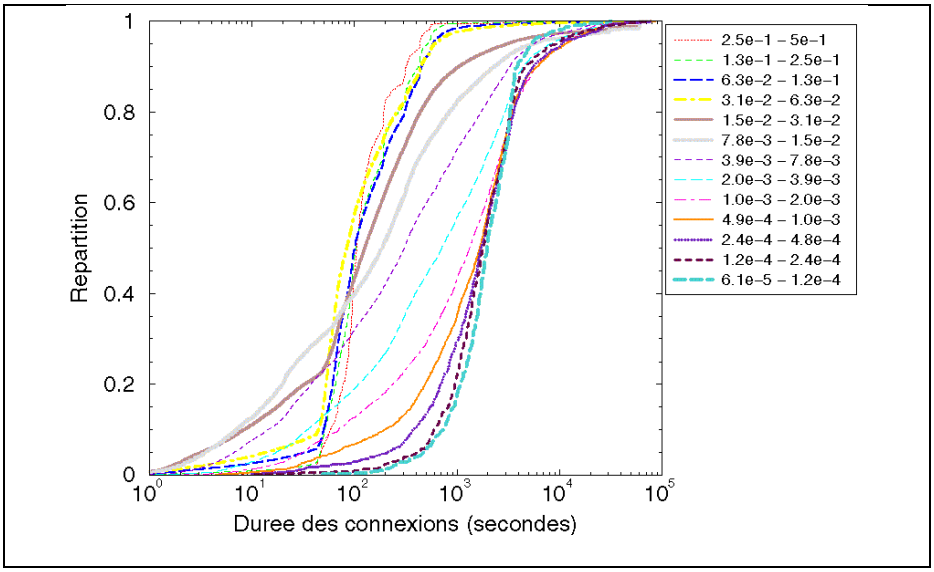


Figure 2. *Distribution des connexions TCP en fonction de leur durée pour divers taux de déséquencements*

3.3. Durée et la taille moyenne des connexions

Nous avons représenté sur la Figure 3 la moyenne de la durée (à gauche) et de la taille (à droite) des transferts en fonction du taux de pertes. Nous observons une légère décroissance de la durée des connexions pour des taux de pertes supérieurs à 10^{-3}, puis une légère croissance pour des taux de pertes de quelques pourcents, entre 10^{-2} et 4.10^{-2}. Pour les taux de pertes supérieurs à 5%, les durées moyennes sont réduites jusqu'à environ 200 secondes, donc d'un facteur 10. Nous observons sur la figure de droite une décroissance continue et beaucoup plus importante, de plusieurs ordres de grandeur, du volume moyen des connexions, surtout pour des taux de pertes supérieurs à 4.10^{-3}. Les courbes sont similaires dans les deux sens montant (Up) et descendant (Down).

En comparant ces deux figures nous constatons la part prépondérante des délais de transmission par rapport aux durées des transferts. En effet, compte tenu de la granularité élevée du temporisateur de retransmission, les délais de retransmission des paquets perdus représentent une part importante de la durée totale des connexions en cas de taux de pertes élevé (Allman et al. 1999), (Padhye et al. 1998). La taille des connexions est beaucoup moins impactée par les paquets retransmis. Il est donc plus pertinent de considérer les corrélations entre le taux de déséquencements et la taille des connexions pour estimer l'influence des performances du réseau sur le comportement des utilisateurs.

Par ailleurs, nous remarquons la valeur importante du coefficient de variation, jusqu'à 10 pour des taux de pertes de plusieurs pourcents. Certaines connexions TCP sont fortement impactées et subissent une proportion élevée de paquets perdus alors que d'autres connexions "passent entre les gouttes" et ne perçoivent pas la surcharge. Une telle dispersion des temps de réponse peut également perturber les clients.

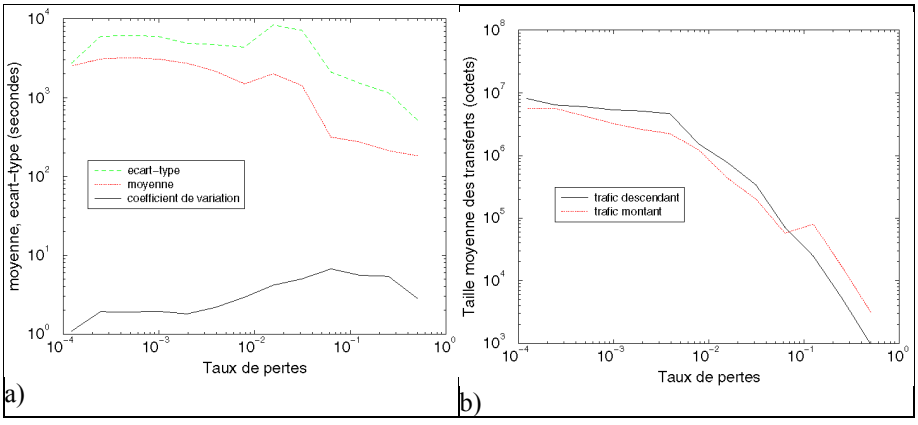


Figure 3. *Durée et taille moyenne des transferts en fonction du taux de pertes*

3.4. Interruptions des connexions

Nous avons noté à la sous-section précédente que la durée et surtout la taille des connexions diminuent lorsque le taux de pertes augmente. Nous allons étudier maintenant l'impact des pertes sur les interruptions de connexions TCP en considérant les drapeaux TCP des derniers paquets échangés. En effet, dans le cas d'une connexion interrompue, l'hôte à l'origine de l'interruption envoie un paquet avec un drapeau Reset (Peukheuri, 2002). Nous avons représenté sur la Figure 4 l'évolution, en fonction du taux de pertes, de la répartition des connexions selon le drapeau de leur dernier paquet. Nous observons une forte proportion de connexions interrompues dès les taux de pertes les plus faibles: par exemple 10% pour 10^{-4} de pertes. Cette proportion augmente ensuite, jusqu'à atteindre un palier à près de 40% de connexions interrompues à partir de 1% de pertes. Au-delà de 20% de pertes, la proportion de connexions interrompues diminue, mais la part des connexions se terminant avec un drapeau S (SYNchronisation) augmente rapidement. Il s'agit des paquets échangés lors de l'établissement des connexions, donc une forte proportion des connexions n'arrivent même plus à s'établir. L'interruption croissante des connexions en fonction du taux de déséquencements peut donc avoir un impact sur la réduction observée de la taille moyenne des connexions TCP dès les plus faibles taux de pertes, dès 10^{-4}, et ce jusqu'à 10^{-2}. Pour des taux de pertes plus élevés, la proportion de connexions interrompues n'évolue plus, la diminution de la taille des connexions pourrait s'expliquer alors par de l'autocensure.

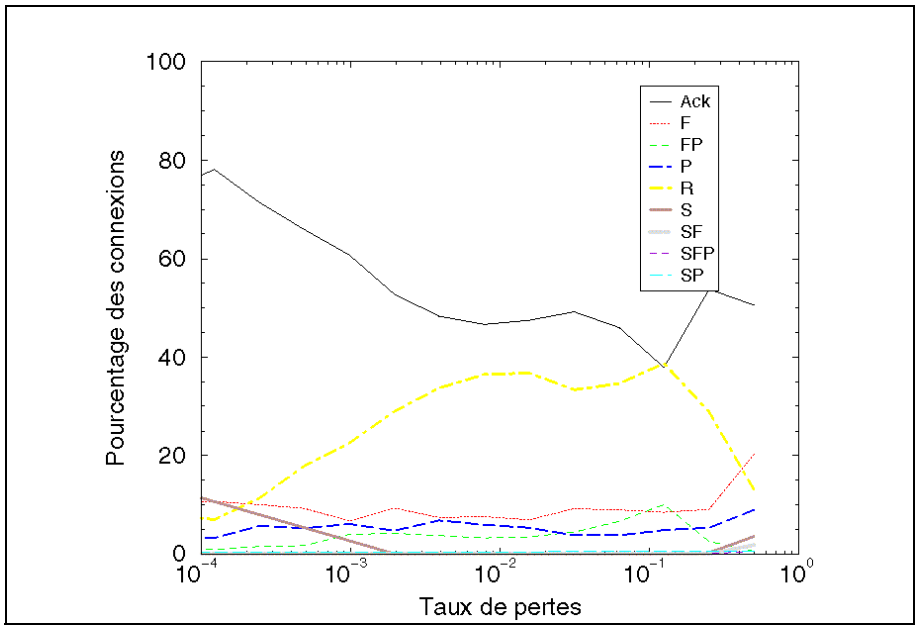


Figure 4. *Répartition des connexions suivant le drapeau de leur dernier paquet, en fonction du taux de déséquencements*

3.5. Taux d'arrivée

Nous étudions dans cette sous-section l'impact du taux de pertes sur l'inter-arrivée des connexions d'un client. A priori plusieurs cas peuvent être envisagés en cas de mauvaises performances. Les utilisateurs peuvent se décourager, et donc espacer leurs transferts. Soit, au contraire, ils peuvent être impatients, arrêter des transferts en cours et relancer des chargements. L'intervalle entre transferts est alors réduit.

Nous avons représenté sur la Figure 5 la moyenne, l'écart-type et le coefficient de variation des inter-arrivées de connexions TCP par client ADSL en fonction du taux de déséquencements de la connexion précédente. Nous observons sur cette figure que l'intervalle entre connexions augmente lorsque le taux de pertes de la connexion précédente croît. D'après la sous-section 3.3 la durée de la connexion précédente diminue dans ce cas. Le silence entre deux transferts croît donc d'autant plus que le taux de pertes du précédent est élevé. Parmi les deux hypothèses contradictoires initiales, nous observons donc plutôt un phénomène de découragement. Par ailleurs nous n'observons pas de seuil avec une croissance de l'inter-arrivée qui apparaît dès les plus faibles taux de déséquencements.

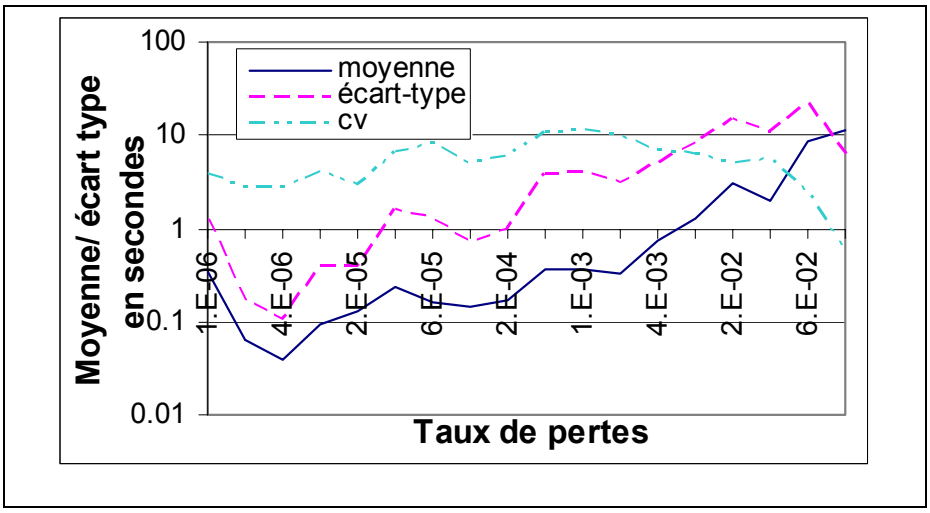


Figure 5. *Intervalle moyen entre deux connexions consécutives d'un client en fonction du taux de déséquencements moyen de la première connexion*

4. Sensibilité de quelques applications aux pertes de paquets

On peut supposer que les applications interactives sont plus sensibles aux pertes. En effet, dans le cas d'un transfert lancé en arrière-plan, l'utilisateur n'attend pas une réponse immédiate. Il aura donc moins tendance à interrompre un transfert qui prend du temps, au contraire d'une application interactive. Par ailleurs, certaines applications, telles que les applications pair-à-pair, génèrent de gros volumes de trafic et s'autorégulent, sans pertes au niveau TCP. Elles peuvent réduire le taux de pertes apparent des grandes connexions. Inversement, d'autres applications générant des transferts courts et souvent simultanés, telles que HTTP, peuvent observer des transferts plus courts sans qu'ils soient interrompus. Les observations précédentes pourraient donc uniquement être dues aux différences entre les caractéristiques de trafic des applications, et non à une influence des pertes sur le comportement des clients. Nous allons montrer dans cette section que les pertes ont bien un impact, quelle que soit l'application, et nous allons essayer de quantifier cet impact.

4.1. Sensibilité aux pertes

Nous essayons de définir ici une méthode générale pour comparer l'impact des pertes sur les caractéristiques de trafic des applications. Nous avons vu qu'il était possible de définir des seuils. Cependant ces seuils dépendent beaucoup du

phénomène que l'on souhaite détecter : une réduction des tailles, une augmentation des interruptions… Une autre possibilité est d'approcher la dépendance de la taille des connexions en fonction du taux de pertes par diverses régressions : linéaire, logarithmique, exponentielle, et en puissance. C'est cette dernière qui présente le meilleur coefficient de corrélation. Le coefficient de corrélation, déjà bon, est encore meilleur si on ne considère que des taux de pertes supérieurs à 10^{-3}. En effet, comme nous l'avons déjà constaté, et en raison du taux de pertes inhérent au fonctionnement à TCP, les taux de déséquencements inférieurs n'ont en général pas d'impact sur les performances des connexions TCP.

Nous trouvons pour le trafic global les régressions suivantes, avec un très bon coefficient de corrélation r:

- Pour le trafic descendant : $y = 3.6 \times^{-1.26}$, $r = -0.99$
- Pour le trafic montant : $y = 7.3 \times^{-1.10}$, $r = -0.99$

Il est possible de la même manière de calculer les régressions de l'inter-arrivée des connexions en fonction du taux de pertes : $y = 13 \times^{0.45}$, $r = 0.95$.

Cette méthode permet de caractériser la sensibilité des caractéristiques de trafic aux pertes de paquets. En effet, la pente est d'autant plus élevée qu'un utilisateur réduit la taille de ses transferts, ou espace ses transferts en cas de taux de pertes élevé.

4.2. Impact des pertes

Suite aux observations de la section précédente, nous ne considérons que l'impact du taux de déséquencements sur la taille en paquets et sur l'inter-arrivée des connexions. Comme la régression en puissance est apparue la plus appropriée pour le trafic global, nous avons uniquement considéré celle-ci pour toutes les applications. Ces pentes sont indiquées dans le tableau 2, pour les principales applications, pour la taille et l'inter-arrivée, avec le coefficient de corrélation associé. Toutes les pentes n'ont pas pu être calculées pour les inter-arrivées en raison d'un nombre insuffisant de mesures. Certaines applications ne sont lancées qu'une fois dans la journée par un utilisateur donné. Nous indiquons également dans ce même tableau la proportion en nombre de connexions et d'octets des applications.

Nous notons tout d'abord une pente non-nulle pour toutes les applications. Donc l'impact des pertes sur la taille des transferts observé pour le trafic global à la section précédente n'est pas un artefact dû aux caractéristiques de trafic des applications. Nous observons ensuite des pentes relativement variées selon l'application, d'autant plus élevées que l'application est sensible aux pertes. Les applications ont été classées en fonction de l'influence des pertes sur la taille des transferts.

tableau 2. Sensibilité des applications aux pertes de paquet

Application	% connexions	%octets	Impact sur la taille	Impact sur l'inter-arrivée

			Pente	Corrélation	Pente	Corrélation
telnet	0.05	0.17	-2.78	-0.97		
edonkey	42.2	67.9	-1.70	-0.96	0.45	0.97
irc	1.7	1.0	-1.55	-0.92	0.20	0.90
citrix	0.04	0.23	-1.52	-0.96		
smtp	0.1	1.2	-0.98	-0.98	0.75	0.98
http	1.9	5.8	-0.91	0.96	0.29	0.95
pop3	0.50	0.3	-0.87	-0.85		
netbios	0.15	0.27	-0.75	-0.69		
ftp	0.20	0.56	-0.75	-0.77		
x11	0.04	0.14	-0.71	-0.94		
bittorrent	0.25	0.62	-0.57	-0.72		

5. Conclusions générales de l'étude

Nous avons mis en évidence et quantifié l'impact du taux de pertes sur diverses caractéristiques des connexions TCP. Cette corrélation n'est pas un artefact dû à des caractéristiques de trafic dépendant de l'heure de la journée ou de l'application. Nous avons vérifié que les caractéristiques globales de trafic varient relativement peu au cours de la journée. Et nous avons observé une influence similaire pour toutes les applications considérées, celles générant le plus de trafic.

Nous avons noté deux seuils sur les taux de pertes, caractérisant leur impact sur les distributions des durées des transferts :

- Des connexions plutôt longues pour des taux de pertes inférieurs à 10^{-3} ;
- Des connexions plutôt courtes pour des taux de pertes supérieurs à 10^{-2} ;
- Un comportement intermédiaire, avec un étalement des distributions des durées, entre ces deux seuils.

En effet, la dispersion des temps de réponse augmente fortement lorsque le taux de pertes dépasse quelques pourcents. Cette variabilité peut être assez gênante pour des applications interactives.

Par ailleurs, les pertes de paquets ont un impact beaucoup plus important sur la taille des connexions que sur la durée. En effet, la durée moyenne des connexions peut augmenter lorsque le taux de pertes augmente, en raison de la part croissante des délais de retransmissions. Pour les taux de pertes observés, les durées varient d'un facteur 10 tandis que les tailles varient de plusieurs ordres de grandeur. Nous avons noté un autre seuil en considérant les tailles, beaucoup impactées lorsque les pertes sont supérieures à 4.10^{-3}. Cependant, la probabilité d'interruption d'une connexion TCP augmente avec les pertes, mais seulement jusqu'à un taux de pertes de l'ordre de 1%. Au-delà, la réduction observée des tailles pourrait s'expliquer par l'autocensure des utilisateurs. Enfin, pour des taux de pertes supérieurs à 20%, une part croissantes des connexions TCP échouent à s'établir.

Considérant l'impact des pertes sur l'intervalle de temps entre les transferts d'un utilisateur, nous avons constaté que les utilisateurs semblent se décourager en cas de mauvaises performances, en espaçant leurs transferts. Nous avons enfin proposé une méthode permettant de caractériser la sensibilité des caractéristiques de trafic aux pertes de paquets. Nous l'avons appliquée pour comparer l'impact des pertes sur la taille moyenne et l'inter-arrivée des connexions de diverses applications.

Nous souhaitons maintenant compléter cette étude en considérant l'influence d'autres critères de performances sur le comportement des utilisateurs : le débit des connexions, le délai de transmission aller-retour… (Rossi, 2002) a en effet observé une influence prépondérant du débit sur la probabilité d'interruption. Nous étudierons ensuite l'impact des performances sur le comportement des clients en fonction de leurs propres caractéristiques. Cet impact doit naturellement dépendre du type d'accès à Internet des applications utilisées par le client, mais probablement aussi de la manière d'utiliser ces applications, taux d'arrivées et tailles pouvant dépendre du client, et des applications utilisées simultanément.

6. Bibliographie

M. Peuhkuri, “Internet Traffic Measurements - Aims, Methodology, and Discoveries”, Licenciate Thesis, Helsinki University of Technology, Finland, 2002

D. Rossi, Ph.D. thesis "At the Network Edge: Sniff, Analyze, Act" Politecnico di Torino, Dipartimento di Elettronica, 2002

F. Guillemin, P. Robert, and B. Zwart. "Heavy tailed M/G/1-PS queues with impatience and admission control in packet networks" IEEE. Infocom'03, San Francisco, CA, March 30 - April 3 2003

J. Roberts, T. Bonald "Congestion at flow level and the impact of user behaviour", Computer Networks 42 (2003) 521-536

S. Yang, G. de Veciana, "Bandwidth sharing: The role of user impatience", IEEE Globecom'01, pp. 2258-2262.

V. Jacobson, C. Leres and S. McCanne, TCPdump, Lawrence Berkeley Laboratory, Berkeley, CA, June 1989.

M. Allman, V. Paxson, W. Stevens, "TCP Congestion Control", RFC 2581, April 1999

J. Padhye, V. Firoiu, D. Towsley, and J. Kurose "Modeling TCP throughput: A simple model and its empirical validation", ACM SIGCOMM'98, Vancouver, CA, September 1998, pp. 303-314.

D. Collange, J.L. Costeux, L. Plissonneau "Detection and Localisation of Performance Limitations of TCP Connections on ADSL", Workshop on QoS and Traffic Control, December 2005, Paris, France

Functional Decoupling Principle Applied to Network Device and QoS Management

romildo martins da silva bezerra, joberto sérgio barbosa martins

Salvador Umiversity
Rua Ponciano de Oliveira, 126
41950-275 , Salvador, BA, Brazil
{romildo,joberto}@unifacs.br

ABSTRACT. *This paper proposes a Police-Based Management Framework based on the functional decoupling principle. The framework is intended to manage QoS-aware devices in complex backbones. The functional decoupling principle corresponds to the isolation and separation of administration police view (rules) from device specific management and device specific control rules and actions. The paper discusses the management principles adopted and the developed framework including its components, module functionalities, some additional supporting tools implemented and, in general, describes the adopted technologies (web, XML, others) in the context of the developed framework.*

RESUMÉ. *Cet article propose un modèle conceptuel (framework) basée sur la technique de gestion basée sur les politiques (PBM) et orienté pour la gestion des réseaux. Le modèle utilise le principe de détachement des visions et politiques (décomposition fonctionnelle) en systèmes (backbones).complexes. Le principe de décomposition fonctionnelle correspond en effet à l'isolation et séparation des visons d'administration (règles) du vison de gestion spécifique des équipements. L'article présent les principes de gestion adoptées et le développement réalisé (outil) en détaillant ses components, les modules fonctionnelles et les technologies adoptées.*

KEYWORDS: *Network Management, Police-Based Management, quality of service, police views, management framework*

MOTS-CLÉS: *Gestion des Réseaux, Qualité de Service, Gestion basée sur les Politiques Décomposition Fonctionnelle - PBM, Vision de Gestion.*

1. Introduction

Computer network device management is a challenging network management sub-area that, effectively, focuses on administration, configuration, operation and control of networking devices in general and routers in particular. Routers, in turn, must be able to support basic packet routing with Quality of Service (QoS) criteria which are fundamental for the new set of networking applications like VoIP (Voice over IP), video and other multimedia applications (Chowdhury, 2001) (Aidarous,2003). In this context, QoS managers must handle a complex set of configuration parameters, configurable devices and specific QoS technologies (token buckets, MPLS - Multiprotocol Label Switching1, explicit routing, constraint-based based routing, and scheduling algorithms, among others) in order to achieve applications requirements.

Concurrent QoS applications are common ground in actual computer network setups2 and, as such, “rules” for concurrent services, “policies” for applying these rules and “decision instances” to reason about rules application under different operational conditions are an essential component in networks supporting QoS. Policy-Based Management (PBM) is based on the indicated principles and is currently presented as a promising solution (Granville, 2001).

PBM is a new paradigm in network management area, offering a solution with flexible characteristics that, moreover, can be automated. Policy-based management allows the creation of highly abstract management infrastructures in which the complexity of the management tasks can be reduced and, as such, an increase in the effectiveness of the network manager activities can be archived.

Various research work are being developed focusing on PBM area (Choi, 2004) (Granville, 2001) (Sloman, 2002) and, among these initiatives, the PONDER project (Sloman, 2002) relates with the proposed implementation by, currently, developing a policy management framework, including tools that control all aspects of policy life cycle. Current research has neither a policy representation standard language nor a storage standard. As such, there is an interoperability issue among solutions requiring specific knowledge and training per framework. Beyond that, specific knowledge is further compromised by the lack of functional decoupling as proposed in this paper.

In practical terms, the proposed management principles adopt PBM as the management strategy and uses “policy database manipulation” as the basic element for the functional decoupling between networks administration vision (functions) and devices support vision (functions). In brief, the principle is to isolate network and

[1] MPLS considered as a technological element supporting QoS strategies.
[2] Exception made for networks where over dimensioning principle is applicable like some ISPs and telecommunications providers with both huge and spare transport facilities.

QoS administration vision from device specific management. It is argued that the proposed decoupling will increase device management effectiveness allowing complex configuration as required for configuring routers in huge backbones supporting multiple QoS-aware applications.

The paper presents in sequence the functional decoupling principle and the framework developed for QoS Management. It is presented the framework components, their functionalities and the use of XML (eXtended Markup Language) (Choi, 2001) (Klie,2004) for policy definition and representation. XML supports information (policy) database syntactic and semantics, uses DTD for syntax verification and considers ontology for supporting formal specification and semantic analysis of the policy database.

2. Functional Decoupling Principle – An Approach for Policy-Based Network Device Management

The functional decoupling between the network administration functions and device support functions, which are tasks typically executed by administrators with different profiles, is intended to provide each manager profile with distinct and appropriate views of the management problem.

The functional decoupling is effectively implemented by splitting the policy database in two distinct partitions - policy database and domain configuration database - and by restricting the framework's module access to these partitions in accordance with the associated manager profile. The policy database defines the policies and references devices and equipments they will act upon. Devices and equipments are referenced only by name (labels) and, as such, all device-related information is hidden from the policy administrator. Device parameters configuration, interconnections, data paths and other network technical aspects are configured and handled by the domain configuration database (Figure 01).

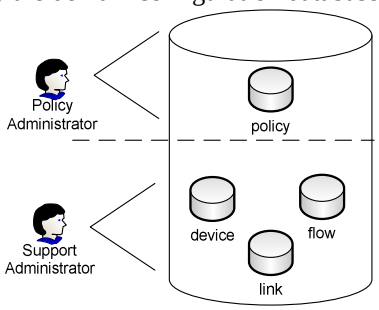


Figure 01. *The Decoupling Principle and Management "Views" for Network Managers*

Managers using customized and functionally decoupled information models for policy and domain configuration database have easier perception of framework's functionalities and have reduced tasks scope, facilitating modules utilization and tools mapping according their specific management profile. The ability to hide data using "views" simplifies the use of the developed framework and, beyond that, improves security as far as users do not need to have access to more information than they effectively need (Figure 01).

3. Policy-Based Management Framework

As indicated, the functional decoupling principle may facilitate device management in huge and complex networks like, for instance, those requiring QoS management and control. In these operational scenarios, it becomes then necessary to specify and implement a "framework" based on the proposed principles that controls the policy life cycle in order to provide managers with an additional management tool. The following sections will then present and discuss the policy-based management framework developed based on the functional decoupling principle.

3.1. *Policy-Based Management Framework Requirements*

The policy-based management framework basic requirement is the efficient management support for all steps and process involved in policy's life cycle. To achieve this, this system provides a set of adequate functionalities and includes supporting tools.

The following additional requirements are also specified and considered in framework's development: "expansibility" (the ability to support new functionalities and/or new policy models creation in the framework); "scalability" (the ability to guarantee scalability and execution of complex policies in complex infrastructures with a large number of equipments and associated policies); "applicability" to multiple and diverse technologies (the ability to take care of to the variety of existing networking technologies); "adaptability" (the ability to take care of the distinct requirements for policy creation); "friendly interface"; and "accessibility" (the ability to allow ubiquitous access through multiple devices or architectures).

3.2. *Policy-Based Management Framework Structure and Management Scenario*

The policy-based management framework is mainly oriented for policy rules creation, administration and maintenance and, primarily, focuses on network QoS Service Level Management. Alternatively, the framework may also be used for managing other functional and operational networking aspects. In practical terms, the

developed framework is composed by 06 modules with specific functionalities as indicated in figure 02.

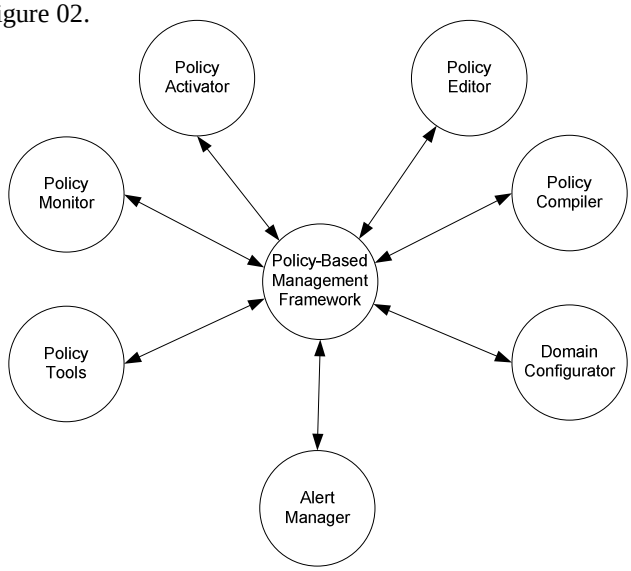


Figure 02. *Policy-Based Management Framework*

The framework itself interacts in a specific management operational scenario composed by a set of additional functional components including a policy repository (database), a policy distribution mechanism, a flow monitor (optional) and a management workstation working as the manager point-of-presence as illustrated in figure 03.

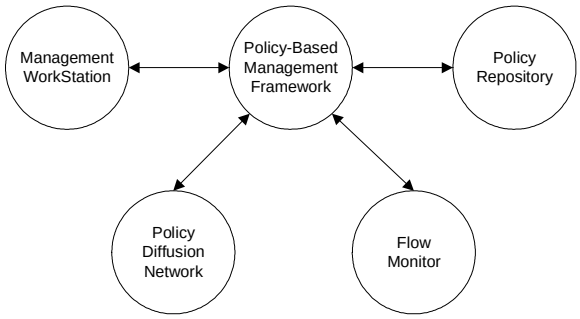


Figure03.*Policy-Based Management Scenario– Functional Components*

The policy-based management scenario components (Figure 03) have specific functionalities as follows:

– Policy-Based Management Framework - It is the developed framework itself and provides centralized administration of network QoS policies. The policy-based management framework interacts directly with the policy

repository in nearly all operations done by its modules (tools). Policy edition, validation and compilation and other auxiliary tools, are available in this framework in order to guarantee the control and management through policies. In brief, the framework is the focal point used by network administrators to interact with the policies and manage QoS.

- Policy Repository – It is the policy database and, additionally, stores other relevant management information as required for the policy-based management approach. Policies are stored in XML and use ontology in order to better support application development.
- Management Workstation – It is the management console through which administrators manage network flows, specify network policies or activate other framework tools. Ubiquitous access to all framework tools and components is a major requirement for the management workstation component.
- Policy Diffusion Network – It is a functional component which, effectively, corresponds to the adopted mechanism for distributing policies through the managed network. The framework does not specify any required policy diffusion mechanism and, as such, any functionally suitable protocol or mechanism may be adopted. COPS-PR (Common Open Policy Service Usage for Policy Provisioning) (Chan, 2001) is suggested as a standardized alternative for policy delivery to network devices (PEPs - Policy Enforcement Points). In case a policy diffusion protocol is adopted, the framework does not have to delivery policies to all network devices but only to those behaving as the equivalent to a PDP (Policy Decision Point), as defined in COPS-PR architecture (Figure 04).

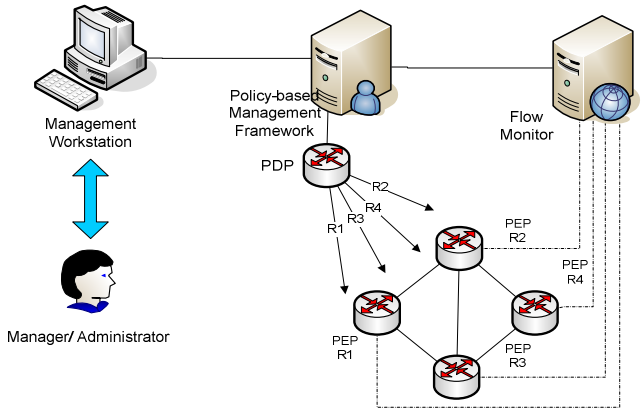


Figure 04. *Policy Diffusion with COPS-PR support*

- Flows Monitor – The objective of this optional component is to continuously evaluate network performance in relation to the policies

defined by network administrators through the framework (Figure 04). The flow monitor component acts fundamentally as service level evaluator in the proposed management scenario. Flows are effectively monitored against the specified service levels.

3.3. *Policy-Based Management Framework Modules*

The policy-based management framework (Figure. 02) supports the creation, edition, validation, compilation and installation of network policies, allows the complete configuration of network domains, provides alerts generation and, moreover, has a set of additional supporting tools applicable, among other alternatives, for policy database maintenance (Bezerra, 2005). Only one station with a browser is necessary for accessing the framework. The developed web interface is friendly and some modules provide assistants for easier configuration process. It follows the modules description and corresponding functionalities.

3.3.1. Policy Editor

The policy editor is the module supporting generic policy definition that is functionally decoupled from the infra-structure (device and equipments) upon which it will be applied. In addition, policy creation does not require any previous knowledge about policy description languages.

Policy creation uses only knowledge about their resources and implementation and configuration details remains hidden. Policies and equipment information are effectively merged through the policy compiler module.

The creation of high level policies and their representation in XML has two intrinsic advantages: allows device technical details abstraction and decouples device administration from policy administration.

3.3.2. Policy Compiler Module

The policy compiler is the module which reads generic policies written in high level language and converts them to either a policy specification language or directly to the technology used by the managed device. In the later case, the policy compiler module makes generic policies applicable to real device and equipments (Figure 05).

The policy compiler associated with XML flexibility supports framework's expansibility. In effect, policies are written in generic form (abstraction), are easily readable (XML) and therefore can not be directly applicable to the managed devices. As such, the policy compiler provides enough flexibility by allowing multiple conversion alternatives suitable for different policy specification languages and technologies.

The developed policy compiler converts for queuing disciplines like CBQ (Class Based Queuing), HTB (Hierarchical Token Bucket), SPSL (Security Policy

Specification Language) and IPTABLES in Linux routers and supports MPLS (MultiProtocol Label Switching) and Cisco's ACL's (Access Control Lists).

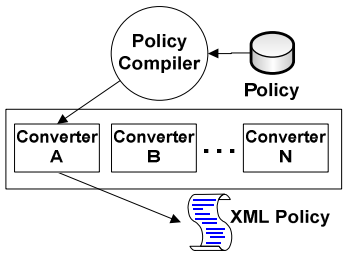


Figure 05. *Policy Compiler Module*

3.3.3. Alert Manager Module

The alert manager is the module dispatching notifications corresponding to programmed events like, for instance, a new policy creation. The alert manager is a key element for policy control and auditing allowing the manager to control the policy modification process for the managed network. In effect, this module is not only an audit component but also an important system security requirement.

3.3.4. Domain Configuration Module

The domain configuration module was developed to support the specification of devices, flows and interconnections in the managed domain. The domain configuration is three steps process: equipments specification, interconnection specification (links) and flow specification. Each of these specifications are separately stored in the corresponding domain configuration database.

The domain configuration module facilitates the distribution of management tasks and attributions. For instance, equipment database specification may be specified by the support staff while flow specification may be specified by domain managers which are responsible for QoS requirements specification for different network flows and applications.

3.3.5. Policy Activator Module

Policies are sent directly to the managed object or to devices with PDP functionality (COPS-PR) by the policy activator module. The policy activator module allows network administrator to send policies by FTP (File Transfer Protocol), a practical solution for most routers, and, additionally, supports sockets for sending policies. It is also possible to activate the policy manually with Telnet or any other alternative of sending policies that the installed PDP supports.

The policy activator module does not search the substitution of the COPS protocol, but instead to guarantee the activation of policies in networks that do not support this protocol.

3.3.6. Other Policy Tools

In addition to the basic modules, the policy-based management framework has an additional set of tools for policy management (transmission, reception and validation of policies) such as:

- Policy Transfer Module – a tool developed to execute the transference of policies between different policy-based management systems.
- Policy Verifier Module – another support tool focusing on policy validation. The validation process occurs by comparing the policy structure (XML) with its DTD (Document Type Definition).
- Policy Receiver Module – this auxiliary module imports policies from another policy-based management system.

The combination of these auxiliary tools allows the transference of policies between policy-based management systems, facilitating the creation of new policies (Bezerra, 2005). The framework also has a tool to backup the policy database (DB backup), a tool that migrates the policy database making it compatible for ubiquitous access through mobile devices (DB for PalmOS) and the tool "PingPolicy" that allows the execution of "pings" in devices related to one specific policy. Currently, the developed framework does not verifies or analyse policy conflicts.

4. Policy Database and XML

XML was the language adopted for representing policies in the policy-based management framework since it complies with the specified system requirements and, also, in order to guarantee a globally accepted representation of data with easy reading in computational terms (Klie, 2004). Anyhow, XML adoption does not guarantee a complete solution, being necessary, for instance, syntax and semantics verification in relation to the policy database and semantic value in relation to stored data. Syntax verification of the XML-based policies uses DTDs to define the construction of valid blocks in XML documents and defines the structure of documents by using a list of valid elements. The policy database attributes specified in XML do not affect meaning to the specified data and, as such, it is necessary to specify and develop an ontology in order to provide an intelligent access structure to the information stored in XML policies and add semantic value to them (Glen, 2002).

In relation to policy database structure, the creation of a unique policy containing information about devices, flows, links and rules are practical, but not recommended

because it affects policy reusability and simplicity (Bezerra, 2005). The proposed solution focus on breaking the given information (policy) by making use of data, device, link and flow information segmentation (Figure 06).

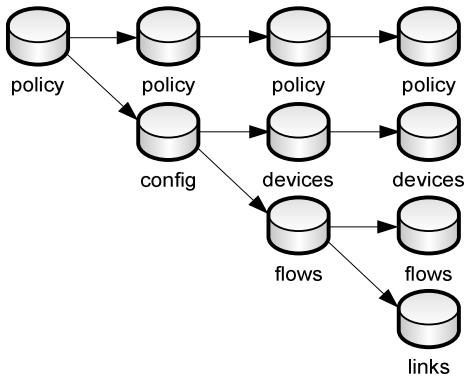


Figure 06. *Policy database structure and segmentation*

As an example, table 01 presents a policy structure based on the net presented in Figure 04.

Policy Example	Rules in CBQ
<?xml version="1.0" ?> - <policy data="21/12/2004" time="14:26:04" administrator="rmartins"> - <node> <origin>R4</origin> <destination>R2</destination> </node> - <rule> - <origin> <ip>10.4.1.1</ip> <mask>24</mask> <port>25</port> </origin> - <destiny> <ip>10.2.1.1</ip> <mask>24</mask> <port>25</port> </destiny> </rule> <bandwidht>64Kbit</bandwidht> - <time> <start>00:00</start> <end>12:00</end> </time> <priority>3</priority> <latency>none</latency> <jitter>none</jitter> <isolated>yes</isolated> <bounded>yes</bounded> <comments>Tráfego SMTP com shaper</comments> </policy>	Router 4 DEVICE=eth0,10Mbit,1Mbit RATE=256Kbit WEIGHT=256Kbit PRIO=3 RULE=10.4.1.0/24:25,10.2.1.0/24:25 TIME=00:00-12:00; 64Kbit/64Kbit BOUNDED=yes ISOLATED=yes Router 2 DEVICE=eth2,10Mbit,1Mbit RATE=256Kbit WEIGHT=256Kbit PRIO=3 RULE=10.2.1.0/24:25,10.4.1.0/24:25 TIME=00:00-12:00;64Kbit/64Kbit BOUNDED=yes ISOLATED=yes

Table 01. *Policy Example in XML and Rule in CBQ.*

5. Final Considerations

The development of the policy-based management framework focused on functional decoupling principle in order to guarantee efficient network device management through the creation, edition, validation, compilation and installation of policies and the use of separate management views for network administrators and device support administrators. In addition, the framework supports easy domain configuration, has an alert manager and supporting tools applicable for policy database maintenance.

The first version of the framework was developed in ASP (Active Server Pages) and currently has migrated for PHP (Personal Home Page). The choice of web technologies guarantees policy-based framework accessibility and availability through a single browser.

The policy-based framework expansibility and scalability are guaranteed by its modular development, the adoption of XML for the policy representation and policy compiler functionality. Database syntax and semantics verification is realized through DTD's, while semantic values assigned to the stored information use a QoS ontology.

One of the contributions of the proposed framework is the adaptable and flexible policy representation with XML. This has favored the concept of "simple" and “legible” policy, which in turn, have functional requirements such as expansibility and scalability. Another positive and innovative point of the framework is the ability to support the functional decoupling management.

In terms of new research actions, there is the improvements of the graphical interface interactivity, the development of new policy converters and the development of a flow monitor.

6. References

Aidarous, Salah; Plevyak, Thomas; Martins, Joberto et all., (2003) “Managing IP Networks – Challenges and Opportunities” John Wiley & Sons/ IEEE Press.

Bezerra, Romildo M. (2005) “Desenvolvimento de um Ambiente Flexível de Gerência de QoS através de Políticas”. Thesis presented to UNIFACS.

Chan, K., et. all. (2001) “COPS Usage for Policy Provisioning”, RFC 3084.

Choi, M., Choi, H. and Hong, J. W. (2004) “XML-Based Configuration Management for IP Network Devices”, In: IEEE Communications Magazine.

Chowdhury, D. D., (2001) “Unified IP Internetworking”, Springer Publisher.

Glen D., Russell L., Ian S. (2002), “Developing an Ontology for QoS” in 5th Annual DIRC Research Conference, pp. 128-132.

Granville, L. Z. (2001) “Gerenciamento Integrado de QoS em Redes de Computadores”. Thesis presented to UFRGS.

Klie, T. and Straub, F. (2004) “Integrating SNMP Agents with XML-Based Management Systems”, In: IEEE Communications Magazine.

Sloman, M.; Lupu, E.; et all (2002) “Tools for Domain-based Policy Management of Distributed Systems”, in Network Operations and management Symposium – IEEE.

SESSION 2

L'AUTO-CONFIGURATION

Gestion autonome de réseaux IP multiservices : auto-configuration et auto-optimisation.

Hajer DERBEL, Nazim AGOULMINE
LRSM (Laboratoire de Réseaux et Systèmes Multimédia)
18, allée Jean Rostand
91025 Evry cedex
derbel@iie.cnam.fr, nazim.agoulmine@iup.univ-evry.fr

Mikael SALAUN
FT R&D
Mikeal.salaun@Francetelecom.com

RÉSUMÉ. L'approche de gestion par politiques a permis aux réseaux de simplifier la configuration de leurs équipements et d'automatiser une partie de leurs processus de gestion. Malheureusement, cette automatisation n'est plus suffisante pour gérer les réseaux du futur qui sont devenus de plus en plus complexes et répartis.
Dans cet article, nous proposons une approche de gestion autonome à base de politiques abstraites (connaissances et objectifs) afin de permettre d'automatiser d'avantage le processus de gestion de ces réseaux. Nous nous intéressons plus particulièrement à la modélisation des connaissances et la conception de l'architecture de l'élément autonome (routeur) à travers l'étude du cas d'un réseau opérateur multiservice. Afin d'expérimenter cette conception et cette architecture, nous avons réalisé un ensemble de tests sur une plateforme de tests. Les résultats ont montré la dynamicité et l'optimisation de la configuration des ressources allouées aux clients, lors de l'accomplissement du but de satisfaire leurs exigences.

ABSTRACT. The policy network management approach allowed to simplify the configuration of network equipments and to automate some parts of their management process. Unfortunately, this automation is not sufficient to manage future networks which become complex and large.
In this paper, we propose an autonomous management approach based on abstract policies (knowledge and objectives) in order to better automate the management process of these networks. We mainly focus on the knowledge modeling and the architecture design of the autonomous element (router) through studying the case of a multi-service operator network. In order to experiment the proposed architecture, we performed several tests on a testbed platform. The results prove the dynamicity and the optimization of the configuration of resources allocated to customers, during the achievement of the goal to satisfy their requirements

MOTS-CLÉS : réseaux autonomes, gestion par politiques, gestion autonome, services.

KEYWORDS: autonomic networks, policy management, self-management, services.

1. Introduction

La dernière décennie a connu une augmentation impressionnante de l'utilisation des réseaux de transmission, notamment le réseau Internet lequel n'avait initialement pour but que d'acheminer au mieux des paquets à leurs destinations. Aujourd'hui, l'utilisation de ce réseau s'est étendue vers le support d'applications multimédias (Voix, Vidéo…) qui sont contraignantes en qualités dites Qualités de Service (QoS).

Afin d'intégrer des mécanismes supportant cette QoS dans le réseau IP, de nombreux travaux notamment à l'IETF ont était menés. Ces travaux ont conduit à la spécification de nombreuses architectures, dont deux ont été standardisées : l'architecture IntServ (Integrated Services) (Wroclawski, 1997) et l'architecture DiffServ (Differentiated Services) (Blake et al, 1998). L'intégration de la QoS dans ces architectures a augmenté la complexité de gestion des réseaux. Dés lors, il n'est plus possible de s'appuyer sur les techniques classiques de configuration et de gestion, basées principalement sur le protocole SNMP (Simple Network Management Protocol) (Case et al, 1990). Ces techniques restent les solutions les plus déployées dans les réseaux fixes et mobiles, par motivation de leur simplicité.

Afin de réduire d'avantage la complexité de la gestion des réseaux offrants de la QoS, une nouvelle initiative de gestion a vue le jour, il s'agit de la gestion des réseaux par politiques (PBM : Policy Based Management) (Wtersinen et al, 2001). L'objectif de cette approche est d'intégrer la gestion et le contrôle de tous les composants du réseau dans un seul système. Ce dernier permet d'appliquer une stratégie de gestion globale (une politique) liée à l'organisation du réseau en s'appuyant sur un modèle d'information sémantiquement riche. En automatisant un ensemble de tâches de gestion, l'approche par politiques a permis de simplifier la gestion des infrastructures des réseaux.

La croissance de ces réseaux en taille et en fonctionnalité rend très complexe la spécification du modèle d'information et des politiques associées. De ce fait et également du fait que l'administrateur est au cœur du processus de la PBM, des problèmes commencent à surgir en termes de cohérence des politiques et de maîtrise de l'expertise globale du système. La solution à ces problèmes ne peut pas venir d'une automatisation partielle mais plutôt totale de la gestion du système et de son processus décisionnel.

La réponse à ce problème d'automatisation totale trouve sa source dans l'initiative de l'Informatique Autonomique (« Autonomic Computing ») (Horn, 2001). En effet, le but de cette initiative est de permettre à un System Informatique (SI) de gérer lui-même les éléments qui le composent et d'entreprendre automatiquement les actions nécessaires à son fonctionnement optimal, sans intervention humaine.

Dans notre travail nous cherchons à adapter les consignes de cette initiative à la gestion des réseaux IP, afin d'automatiser le processus de leur gestion et distribuer la prise de décision sur leurs composants.

Dans cet article, nous proposons une approche de gestion autonome à base des connaissances et des objectifs. On s'appuie pour cela sur la PBM qui, malgré ses limites, reste l'approche la plus intéressante pour la mise en place de processus décisionnels automatiques. Nous nous intéressons plus particulièrement à la modélisation des connaissances et à la conception de l'architecture de l'élément autonome (routeur) à travers l'étude du cas d'un réseau multiservice. Afin d'expérimenter cette approche, nous avons réalisé un ensemble de tests sur une réalisation matérielle. Les résultats de ces tests ont montré une configuration dynamique et optimisée des ressources allouées aux clients, lors de la réponse au but de l'administrateur qui se résume dans la satisfaction maximale des exigences des clients.

Le reste de cet article est structuré de la manière suivante : après la description de notre motivation de travail dans la section 2, la section suivante décrit l'essence de la gestion autonome des réseaux. Dans la section 4, nous présentons notre approche de gestion autonome. La section 5 permet de décrire la représentation de connaissances et l'architecture du routeur autonome dans le cas d'un réseau multiservice. L'expérimentation de l'approche proposée et les résultats obtenus sont présentés dans la section 6. Enfin, la section 7 conclut cet article.

2. Motivation du travail

Aujourd'hui, les approches de gestion classiques ne permettent pas d'atteindre les objectifs des opérateurs. L'approche de gestion qui était prometteuse, approche par politiques, a également montré des limites. En effet, cette approche a été conçue pour faciliter la gestion des réseaux par l'utilisation des règles dites règles de "politiques". Ces dernières sont décrites sous forme de clause (**si conditions alors actions**). En centralisant les informations et la prise de décision sur ces politiques dans un serveur dit serveur de politiques ou PDP (Policy Decision Point), la gestion est effectuée à travers la communication entre ce PDP et les points d'application de politiques dits PEPs (Policy Enforcement Point) (Agoulmine et al, 2002). Cette communication est rendue possible grâce au protocole COPS (Common Open Policy Server) (Durham et al, 2000), protocole de transport d'objets de politiques. L'architecture de la PBM comporte également une console d'administration qui permet au gestionnaire notamment d'éditer les politiques.

Les besoins des utilisateurs peuvent changer fréquemment et d'une façon inattendue. Face à cette situation, il est impossible pour un réseau opérateur, géré par des politiques normales, de prendre en compte ces changements dans un délai raisonnable. En effet, ces changements nécessitent des modifications, parfois importantes et souvent complexes, des règles de politiques et surtout de leurs parties "action" concernant la configuration des équipements. Faites par un gestionnaire humain, ces modifications peuvent entraîner d'importants problèmes en cas d'erreurs et même une panne complète du réseau que l'opérateur ne peut pas supporter.

Par conséquent, la solution pour les opérateurs est d'introduire plus d'automatisation dans le processus de gestion de leurs réseaux, évitant ainsi les interventions humaines, et rendant le réseau de plus en plus « autonomic ». L'idée est que l'administrateur n'introduit que des objectifs de haut niveau (buts) (Bandara et al, 2004) (Beigi et al, 2004) (Bearden et al, 2001) et que le système interprète automatiquement ces objectifs, et les applique après leur transformation en politiques. Il s'agit d'une vision de la gestion autonome des réseaux.

3. Essence de la gestion autonome des réseaux

Ces dernières années, la communauté informatique a vécu un grand mouvement par le lancement de l'initiative de l'informatique autonomique par IBM en 2001 (Horn, 2001). L'essence de cette initiative est la gestion autonome des systèmes. En effet, cette initiative est définie comme "le modèle de programmation dans le quel le système est capable de s'auto-gérer : s'auto-configurer, s'auto-réparer, s'auto-proteger et s'auto-optimiser" (Ganek et al, 2003). En outre, un SI est géré d'une façon autonome s'il se configure et se reconfigure dynamiquement, optimise l'utilisation de ses ressources et se protège contre les attaques tout seul.

En ce qui concerne les réseaux, automatiser la gestion revient à libérer le gestionnaire de tout détail de l'exploitation et de l'entretien du système, et de fournir aux utilisateurs des services qui répondent à leurs demandes (Strassner, 2003). En effet, un réseau autonome est un réseau qui parvient à offrir un service avec les qualités souhaitées en s'auto-configurant et s'auto-gérant sans intervention de l'administrateur ou d'un système de gestion externe. La conception de l'architecture d'un tel réseau forme la problématique de nombreux travaux de recherche.

Dans ces travaux de recherche, l'autonomie des réseaux est étudiée suivant différents points de vue. Du point de vue de l'utilisateur, le réseau est autonome s'il répond à ses intentions et ses souhaits en termes de qualité (QoS, disponibilité, sécurité, coût). En effet, les utilisateurs souhaitent avoir un accès à des services personnalisés tout en exigeant la qualité pour leurs services préférés (Pujolle, 2005). Chargé de garantir les qualités demandées par ses clients, l'opérateur défini l'autonomie de son réseau par l'utilisation des mécanismes adaptatifs et autonomes pour la gestion du réseau et le contrôle de ses services (Melcher et al, 2004).

Le réseau est autonome, de notre point de vue, s'il est capable de s'auto-gérer en fonction des objectifs de son administrateur, des demandes de ses utilisateurs et de son état. La conception de l'architecture d'un tel réseau (particulièrement : auto-configurable et auto-optimisable) forme notre objectif principal. Pour aboutir à une telle architecture, nous allons d'abord présenter l'approche générale de gestion autonome à base de connaissances abstraites et d'objectifs. Ensuite nous allons décrire cette approche dans le cadre de la gestion d'un réseau multiservice. Nous intéresserons particulièrement à la description des connaissances et de l'architecture du routeur autonome et nous expliquerons le comportement autonomic de ce dernier.

4. Proposition d'une approche de gestion autonome à base des connaissances et des objectifs

Aujourd'hui, la gestion des systèmes de communication a besoin de plus d'autonomie et de décentralisation. Afin de pallier à ces problèmes, nous proposons une nouvelle approche de gestion à base des connaissances. Le but de cette approche est de concevoir un réseau capable de s'autogérer d'une telle manière que le comportement de chaque élément satisfait le but global du système. En effet, notre proposition permet la distribution de la prise de décision sur les différents éléments du réseau notamment les routeurs. Ces derniers, dans cette proposition, sont devenus plus autonomes, par l'automatisation de leurs processus de gestion interne, et plus coopératifs, par la communication et l'échange des informations entre eux.

Comme le montre la figure 1, l'architecture générale de notre approche de gestion décrit deux couches :

- **Couche de définition des connaissances et des buts :**

Cette couche permet de définir les objectifs de l'opérateur ainsi que l'ensemble des connaissances nécessaires à la description abstraite des comportements des composants. Elle est composée essentiellement de deux éléments : un administrateur et un serveur de connaissances. L'administrateur définit l'ensemble de ses objectifs de haut niveau. Ces objectifs seront transmis au serveur de connaissances. Ce dernier fournit les outils nécessaires à la transformation des objectifs en connaissances, à la gestion des SLAs et au transfert des connaissances aux équipements.

- **Couche d'application des connaissances :**

Cette couche représente l'ensemble des équipements capables de s'auto-gérer en adaptant leurs comportements aux buts qui leurs sont imposés et en fonction des connaissances qui leurs sont transmises. Ces équipements forment une commune, dans laquelle, ils communiquent et coopèrent afin de générer un comportement global et optimal qui satisfait tous les objectifs.

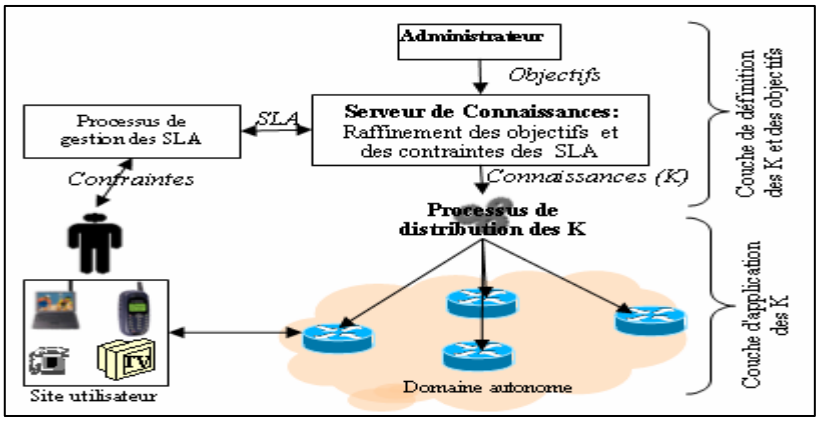


Figure 1. *Approche de gestion à base des connaissances et objectifs.*

En adoptant une approche d'étude de bas en haut (bottom-up), nous avons commencé la réalisation de cette approche par la conception de l'architecture des routeurs et la représentation des connaissances. Afin de mieux décrire ces éléments nous étudierons le cas de gestion d'un réseau opérateur multiservice.

5. Gestion du réseau opérateur multiservice

Dans le cadre d'un réseau opérateur multiservice, l'approche de gestion proposée vise à permettre aux équipements réseau de gérer d'une façon dynamique et optimale leurs ressources en fonction de l'ensemble des demandes et exigences des clients.

La négociation et l'établissement d'un contrat (SLA[1]) entre le client et l'opérateur de services forment les premières étapes. Ce SLA permet de spécifier la bande passante garantie pour le client ainsi que l'ensemble des services demandés (SERi) classifiés par un indice de qualité (QoSi). Cette spécification aboutit à la génération d'un ensemble de connaissances sur les politiques (K-policy) relatif à ce client et ses services préférés. Ce K-policy est enregistré, par la suite, dans la base de connaissances du serveur de connaissances. Le processus de distribution de connaissances transmet ce K-policy à tous les routeurs concernés[2]. En fonction des connaissances transmises (K-policy), les routeurs gèrent d'une façon autonome et coopérative les flux de l'utilisateur indépendamment de leur serveur.

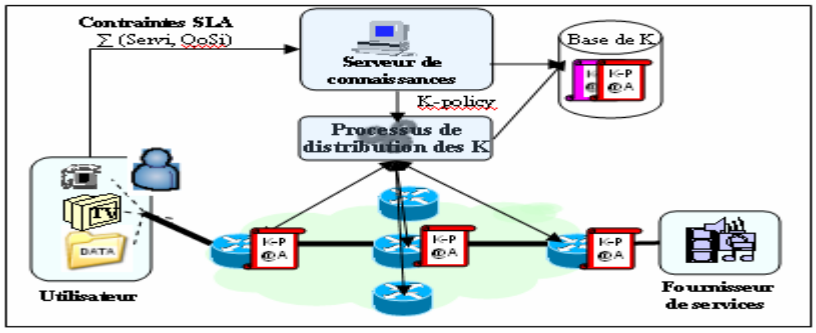


Figure 2. *Gestion du réseau multiservice à base des connaissances.*

5.1 K-policy : connaissances sur les politiques

Le K-policy est un ensemble de connaissances relatif à un client. Il est composé de deux parties : une partie déclarative concerne les connaissances relatives au client et à ses services, et une partie descriptive qui décrit le comportement des routeurs.

---

[1]. SLA : Service Level Agreements.
[2]. La désignation de l'ensemble de s routeurs par le processus de distribution de connaissance ne sera pas décrite dans cet article.

Les connaissances déclaratives permettent d'identifier l'ensemble des spécifications des services demandés et la bande passante (BW) garantie pour le client. Chaque service est décrit par un identificateur (SERi), un ensemble de critères de filtrage (Fi= {type des flux, protocole, port destination, port source, adresses IP}), une bande passante nécessaire (BWi) et un indice de qualité (QoSi= {1, 2, 3,..}) décrivant l'importance de ce service (priorité).

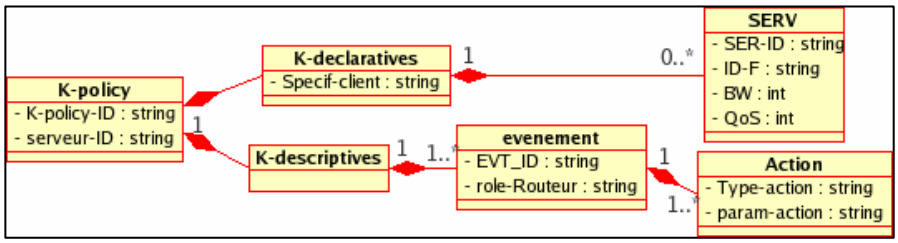


Figure 3. *Spécification du K-policy*

Les connaissances descriptives du comportement du routeur permettent de décrire les événements qui peuvent se produire au niveau de chaque routeur suivant son rôle dans le réseau (de bordure ou de cœur). Les événements peuvent être internes ou externes, applicatifs (identification d'un client, détection d'un nouveau flux, déconnexion d'un client), ou reliés au réseau (insuffisance de bande passante, taux de perte, etc). Chaque événement est décrit par son identificateur (EVTi= {connexion client, détection de flux, déconnexion client, taux de perte,…}) et un ensemble de paramètres. Il fait appel à un ensemble d'actions (ACTi). Chaque action est décrite par son type (Configuration, Monitoring, Communication...) et un ensemble de paramètres dont les valeurs sont partiellement définies. La spécification entière de ces valeurs est accomplie de manière locale et autonome par les routeurs. Ces derniers jouent un rôle primordial dans l'approche de gestion proposée. En effet, le comportement autonomic et la participation à la prise de décision de chaque routeur permettent d'aboutir à une première forme de réseau autonome.

5.2 Architecture du Routeur Autonome (RA)

Dans l'approche de gestion que nous proposons, le routeur est autonome. En effet, il possède des capacités lui permettant de se gérer et de communiquer avec ses voisins à partir des K-policy. En s'inspirant de (Kephart et al, 2003), l'architecture de notre routeur est composée de différents modules :

- un module de monitoring (collection des données de contrôle).
- un module d'analyse (analyse des données collectées).
- un module de planification (création des plans d'action).
- un module de configuration (commande l'exécution de l'ensemble des actions).

En plus de ces modules, nous envisageons pour chaque routeur un ensemble de capacités de raisonnement pour calculer les paramètres nécessaires pour son processus de gestion et des capacités de communication permettant d'acheminer les connaissances entre les différents composants et leur serveur en cas de besoins.

Comme le montre la Figure 4, à l'intérieur du routeur et en fonction des connaissances transmises (K-policy) par le serveur de connaissances, une boucle de gestion locale se développe. Plus précisément :

- Le module de Monitoring est chargé d'exécuter les actions de Monitoring, il permet d'implanter les processus de monitoring relatifs à ces actions et en fonction de l'état du réseau. Le suivi des résultats de ces processus permet la détection des signes d'apparition d'événements (symptômes d'événements).
- Le module d'Analyse permet l'analyse des symptômes détectés et des données récoltées. Cette analyse aboutie à une spécification de l'événement produit. En coordination avec le module de Raisonnement, l'Analyseur envoie une demande de changement au module de planification. Les changements peuvent être des changements de la configuration ou des processus de monitoring ou de communication.

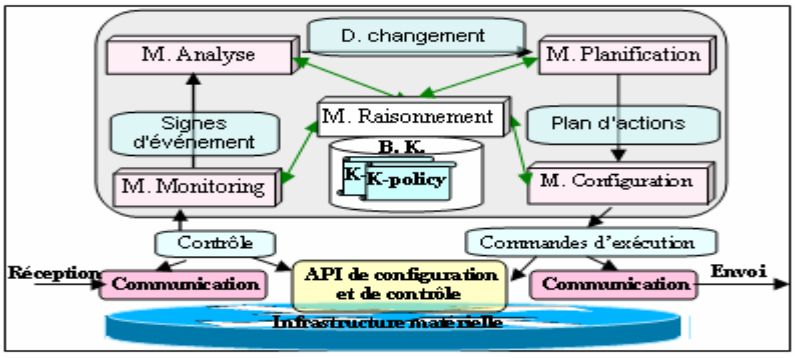


Figure 4. *Architecture du routeur autonome*

- Le module de planification permet la spécification des règles de politiques nécessaires pour effectuer les changements désirés. Il fait appel au module de raisonnement pour calculer les différents paramètres nécessaires. Une fois qu'il a reçu ces paramètres, il organise les règles en créant un plan d'actions pour le module de configuration et/ou un autre pour le module de communication.
- Le module de configuration permet de transformer les règles de politiques en appel de commande d'exécution. Ces commandes peuvent être des commandes de communication ou de configuration.
- Le module de communication permet de gérer les deux sens des communications (envoi et réception) du routeur avec son serveur et ses voisins.

En effet, il permet : 1) de contrôler les événements externes provenant du serveur ou des routeurs voisins (réception), 2) d'envoyer des messages de notification aux voisins et à son serveur (envoi).

La boucle de gestion du routeur se termine par l'exécution de l'ensemble des commandes de configuration (en utilisant les API de configuration et de contrôle[3]) et de communication.

6. Plateforme et résultats de tests

Dans cette section, nous décrivons notre plateforme de test et nous présentons les résultats des tests réalisés sur cette plateforme.

L'objectif principal de la réalisation de cette plateforme est de montrer la gestion dynamique et automatisée au niveau des routeurs afin de tester l'approche de gestion proposée dans le cas d'un réseau opérateur multiservices. Afin de satisfaire les demandes et les préférences des clients, l'approche de gestion proposée tente à automatiser la configuration optimale des routeurs et l'adapter aux demandes du client sans intervention de l'administrateur en utilisant des K-policy au lieu des politiques simples.

6.1 Description de la plateforme

La plateforme de test est constituée d'un réseau IP représentant le réseau de l'opérateur. L'étude de cas de cette plateforme met en œuvre un client (A), un serveur de connaissances, deux routeurs[4] et un fournisseur de service multimédia (figure 5).

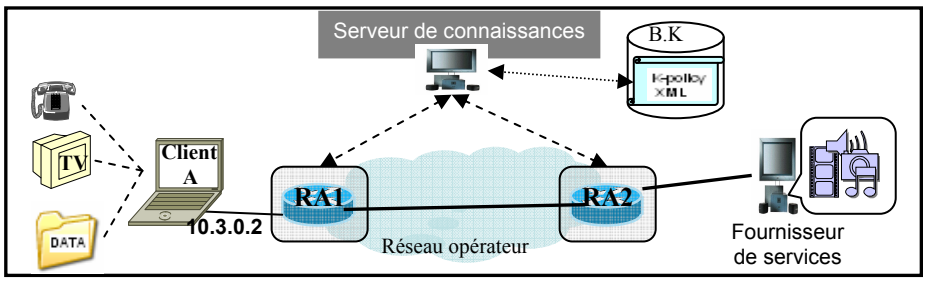


Figure 5 : *architecture de tests*.

Dans le cadre des tests de cette plateforme, la négociation et l'établissement du contrat du client ainsi que la transformation de ce contrat en K- policy ne seront pas

3. Afin de faciliter la configuration et la rendre plus transparente, nous avons développé des APIs. Ces APIs permettent une configuration simple et efficace en minimisant le nombre de paramètres en faisant appel à des scripts TC (trafic contol sous linux).
4. Les routeurs sont des PCs, sous linux, ayant chacun trois interfaces.

traitées. Nous supposons que le client a signé un contrat spécifiant la bande passante garantie (BW= 2300Kbits/s) et l'ensemble des services demandés ainsi que leurs indices de qualité (comme l'illustre le tableau 1). Par cette spécification le client A demande que la téléphonie doit avoir une haute qualité (QoS=1), les vidéos doivent avoir une bonne qualité (QoS=2). Le tableau 2 décrit le résultat de la transformation de cette spécification par le serveur de connaissances.

SERi	Téléchargement via TCP	Vidéo 1	Vidéo 2	téléphonie
QoSi	3	2	2	1

Tableau 1. *Services demandés par le client A.*

SERi	QoSi	Fi	BWi
TCP (SER1)	*3*	@dst A= 10.3.0.2 protocol= 6	-
Vidéo (SER2)	*2*	@dst A= 10.3.0.2 protocol=17 dport = 4000	800
Vidéo (SER3)	*2*	@dst A= 10.3.0.2 protocol=17 dport = 3000	500
VoIP (SER4)	*1*	@dst A =10.3.0.2 protocole=RTP	64

Tableau 2. *Spécification des services dans le K-policy*

Dans le cas de cette réalisation, nous décrivons les événements suivants :

EVT	Spécification événement
Initialisation	événement d'initialisation.
NewSer	événement déclenché quand le routeur de bordure détecte la présence d'un nouveau service sur son interface d'entrée.
Nmessage	événement déclenché quand le routeur reçoit un nouveau message.
Restart	événement de réinitialisation

Tableau 3. *Les événements produits dans le cadre de tests effectués*

Pour prendre en charge ces événements, le RA peut effectuer les actions suivantes :

Type d'action	Spécification action
Configuration	Les actions de type configuration permettent un ensemble de commandes de configuration dans les routeurs : classification, marquage, création de classes de service, lissage, filtrage, répartition de la bande passante, etc.
monitoring	monitor Nouveau service, monitor réception de message, monitor événement réseau...
Message	Action d'envoi de messages au routeur voisin ou au serveur de K.

Tableau 4. *Les actions produites dans le cadre des tests effectués*

L'ensemble des événements ainsi que les actions correspondantes sont décrits dans le K-policy, en plus des données déclaratives relatives au client (tableau 2).

6.2 Résultats des réalisations

Afin de suivre les événements produits lors de la réalisation de l'approche de gestion proposée, nous examinons un scénario de test. Nous commençons ce scénario par le transfert du K-policy aux routeurs. Dés ce transfert, les deux RA

initialisent leurs configurations (création de la classe de service par défaut BE pour RA1 (qui joue le rôle d'un edge) et réservation d'une bande passante égale à 2300 au niveau de RA2 (qui joue le rôle d'un core)). Par la suite, RA1 implante le processus de contrôle de la présence des services sur son interface. Le lancement du SER1 (prio=3) ne change pas ces configurations vu que ce service est non prioritaire. La diffusion de la vidéo (SER2) déclenche un événement (NewSer) au niveau de RA1. Comme réponse à cet événement RA1 décide de : 1) créer une nouvelle classe de service EF afin de pouvoir différentier les deux services ayant des indices de qualité différents, 2) créer le filtre adéquat au service détecté, 3) envoyer un message de notification à RA2 en l'informant de la présence du SER2 et qui sera marqué en EF. Dés la réception de ce message, RA2 met à jour la répartition de la bande passante réservée au client A en affectant à la classe EF une bande passante égale à celle nécessaire au SER2 (c.à.d 800Kbit).

En parallèle, RA1 surveille de nouveau la présence de nouveaux services. La détection du flux VoIP (SER4) déclenche de nouveau l'événement (NewSer). Pour répondre à cet événement, RA1 calcule de nombre de classes nécessaires pour différentier tous les flux relatifs aux services lancés. Ce calcul lui permet de décider de : 1) créer une nouvelle classe de service AF, 2) changer le marquage du SER2 en AF par la mise à jour du son filtre, 3) créer le filtre adéquat au service récemment détecté, 3) envoyer un message de notification à RA2 pour l'informer de la présence du SER4 qui sera marqué en EF et du changement de la classe de service pour SER2 qui sera marqué en AF. RA2, dés la réception de ce message, met à jour la répartition de la bande passante en affectant à a classe EF une bande passante égale à celle nécessaire au SER4 (64Kbit) et à la classe AF une bande passante égale à celle nécessaire au SER2 (800Kbit). Cette répartition serait égale à 64Kbit pour EF, 1250Kbits pour AF et 986Kbits pour BE après la connexion du SER3 en présence des trois autres services. Cependant, si le client A change l'ensemble de ses préférences au cours de sa connexion, RA1 informe RA2 ainsi que le serveur de connaissances de ce changement.

Ce scénario a permis de décrire l'optimisation et le dynamisme de la configuration dans le processus de gestion de chaque routeur. En plus, cette configuration est adaptative à la classification des services et à l'ordre de lancement de ces services. En effet, la répartition de la bande passante et la création des classes de services se font en fonction de la déclaration des services et l'ordre de leurs apparitions dans le réseau.

La figure 6 illustre la répartition de la bande passante sur les classes de services au niveau de RA2, pour 4 scénarios de lancement des services, sachant que QoS(SER1)=3, QoS(SER2)=2, QoS(SER3)=2, QoS(SER4)=1.

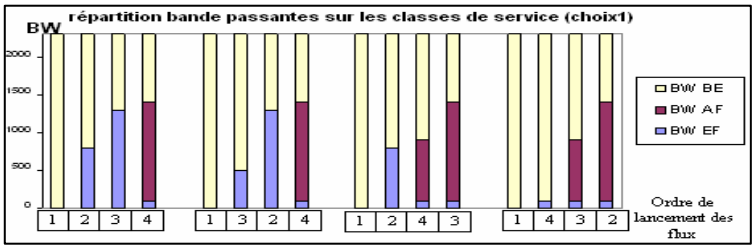


Figure 6. *Répartition de la bande passante sur les classes de service (1)*

En changeant les indices de QoS de cette façon : QoS(SER1)=3, QoS(SER2)=1, QoS(SER3)=2, QoS(SER4)=2 et en suivant les mêmes ordres de lancement des flux relatifs aux services demandés par le client que précédemment, la répartition de la bande passante sur les classes de services change. La figure suivante montre les nouvelles répartitions pour l'ensemble des scénarios.

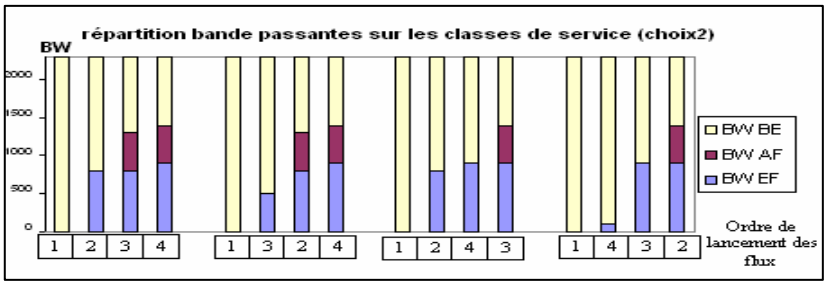


Figure 7. *Répartition de la bande passante sur les classes de service (2)*

L'ensemble de ces tests démontre l'automatisation de la configuration optimale et adaptative des routeurs en fonction des connaissances qui lui sans transmises.

7. Conclusion

Dans cet article, nous avons proposé une approche de gestion autonome, basée sur des politiques abstraites, adaptée aux réseaux IP multimédia pour le support de la QoS de bout en bout. Cette approche permet d'automatiser la gestion des ressources à l'intérieur de chaque équipement afin de satisfaire les objectifs de l'administrateur. Afin de réaliser cette proposition, nous avons commencé par concevoir un modèle de connaissances sur les politiques et une architecture pour les routeurs autonomes.

Afin de tester ce modèle et cette architecture et de démontrer l'automatisation du processus de gestion à l'intérieur du routeur, nous avons mis en place une plateforme de test. Le processus de gestion autonome au niveau de chaque routeur a permis

principalement l'optimisation et le dynamisme de la configuration des ressources et l'adaptabilité de cette configuration au choix des services et à l'ordre de leurs apparitions dans le réseau.

Les travaux futurs visent à étendre l'étude de l'approche de gestion proposée afin de permettre de plus de coopération entre les éléments du réseau de telle manière que le comportement global de tout réseau satisfait les buts de niveau élevé. En plus, nous cherchons à étudier les événements qui peuvent se produire dans les réseaux à grande échelle et à étendre les fonctionnalités des APIs de configuration et de contrôle dynamiques pour permettre la mise à jour dynamique des différents paramètres des files d'attentes afin d'optimiser et améliorer les performances de bout en bout.

8. Bibliographie

Agoulmine N., Cherkaoui O., "La Pratique de la Gestion de Réseaux ", ouvrage, page 272 , ISBN: 2-212-11259-9, Mars 2002.

Bandara A. K., Lupu E. C., Moffett J. D., Russo A., "A goal-based approach to policy refinement", IEEE, POLICY 2004, pages 229–239, Juin 2004.

Bearden M., Garg S. and Lee W., "Integrating Goal Specification in Policy-based Management", Policy Workshop, USA, Janvier 2001.

Beigi M.S., Calo S., Verma D., "Policy Transformation Techniques in Policy-based Systems Management", IEEE, POLICY 2004, Juin 2004.

Blake S., Black D., Carlson M., Davies E., Wang Z., Weiss W., "An Architecture for Differentiated Services", RFC 2475, Décembre 1998.

Case J., Fedor M., Schoffstall M.L., Davin C., "Simple Network Management Protocol (SNMP)", RFC1157, Mai 1990.

Durham D., Boyle J., Cohen R., Herzog R., Rajan R., Sastry A., "The COPS (Common Open Policy Service) Protocol", RFC 2748, Janvier 2000.

Ganek A. G. and Corbi T. A., "The Dawning of the Autonomic Computing Era", IBM Systems Journal 42, No. 1, 5-18, 2003.

Horn P., "Autonomic Computing: IBM's Perspective on the State of Information Technology", IBM Corporation, Octobre 2001.

Kephart J.O., Chess D., "Vision of Autonomic Computing", Computer Magazine, IEEE, 2003.

Melcher B. et Mitchell B., "Towards an Autonomic Framework: Self-Configuring Network Services and Developing Autonomic Applications", Intel Technology Journal, Novembre 2004

Pujolle G., "Les réseaux 2005", ISBN : 2-212-11437-0, France 2004.

Strassner J., "Realizing on demand networking", MMNS 2003, 6th IFIP/IEEE International Conference on Management of Multimedia Networks and Services, Queen's University of Belfast, Irlande du Nord, September 2003.

Wroclawski J., "The Use of RSVP with IETF Integrated Services", RFC2210, Septembre 1997.

Wtersinen A., Schnizelein J., Strassner J., Scherling M., Quinn B., Perry J., Herzog S., Huynh A., Carlson M., Waldbusser S., "Policy terminology", Internet Draft, draft-ietf-policyterminology-02.txt, Mars 2001.

Un Outil Générique de Gestion par Politiques: Validation de l'Implémentation dans un réseau sans fils WLAN*

Ana Luiza B. de P. Barros, Joaquim Celestino Jr, Laure Waha N. Mendouga, Marcial P. Fernandez, André Luís de O. Campos, Daniel P. Vieira, Felipe C. Torres, Fernanda Lígia R. Lopes, Francisco Wagner C. Aquino, Marcel Maurício F. De Alencar, Rafael R. Soares

Laboratório de Redes de Comunicação e Segurança (LARCES)
Universidade Estadual do Ceará (UECE)
Av. Paranjana, 1700 – 60740-000 – Fortaleza, CE – Brasil
{analuiza,celestino,laure,marcial,andre,daniel,felipecolares,fernanda, wagner, marcel,rafael}@larces.uece.br

RESUME: *La Gestion par Politiques a été une architecture très utilisée pour configurer et offrir la QoS dans les réseaux qui aujourd'hui sont construits de plus en plus d'équipements différents et de divers fabricants. Dans cet article, est présenté le LARCES_PBM, un framework générique de gestion par politiques implémenté pour offrir la QoS dans n'importe quel type de réseaux. Comme étude de cas, il a été implémenté un prototype pour les réseaux sans fils WLAN. A partir d'un scénario de tests proposé, il a été prouvé l'application correcte des politiques dans ce type de réseaux, ce qui implicitement a démontré les fonctionalités de l'architecture implémentée.*

ABSTRACT. *Policy-Based Management has been a widely used architecture to configure and to provide QoS in networks which today are build more and more with differents devices and from varied manufacturers. This paper presents LARCES_PBM, a policy-based generic framework designed to provide QoS in any networks types. As study case, an prototype has been developed to WLAN wireless network management. From a considered testbed it was proved the correct policies application in this network type, which has implicitly demonstrated the functionalities of implemented architecture.*

MOTS-CLÉS : *PBMN, framework, WLAN*

KEY-WORDS: *PBMN, framework, WLAN*

* Travaux realisés avec les ressources de HP Brasil P&D referents à la Loi nº 10.176 (Lei de Informática).

1. Introduction

Les premières solutions de gestion de réseaux étaient orientées vers la monitoration et la configuration individuelle de chaque dispositif; Celles-ci sont devenues impraticables ou hautement exhaustives, dues à la croissance des réseaux, à la diversification des types de données transportés, lesquels possèdant des exigences différentes en termes de qualité de service et de sécurité.

Les opérateurs de télécommunications et les administrateurs de réseaux depuis lors, veulent automatiser le processus de configuration des noeuds du réseau. Cette automatisation a pour objectif de mieux controler les trafics qui transitent, tentant offrir la QoS nécessaire et faciliter la gestion des équipements. Ainsi, il a été crée un nouveau paradigme, standardisé par l'IETF/DMTF (*Internet Engineering Task Force* and *Distributed Management Task Force*), visant la couverture de ces problèmes, dénommé Gestion de réseaux par politiques (*Policy-Based Network Management* – PBNM). La gestion de réseaux est dorénavant vue à partir de la vision de services, via SLAs (*Service Level Agreements*).

Le laboratoire LARCES a implémenté un *framework* générique appelé LARCES_PBM, basé sur le modèle proposé par l'IETF/DMTF, pour la gestion par politiques couvrant la qualité de service dans divers types de réseaux, cablés ou sans fils, où il suffit à peine de l'adapter à chaque type. Ce travail a contribué à l'implémentation du *mapping* du modèle d'information PCIM (*Policy Common Information Model)* au schéma de stockage utilisé dans le répertoire de politiques, l'introduction d'une nouvelle manière de manipuler les PIBs (*Policy Information Base*), et d'une forme simplifiée de communication entre certains modules de l'architecture. Comme étude de cas, le LARCES a développé un prototype orienté vers la gestion de réseaux sans fils WLAN. En effet, dans ce type de réseaux, par exemple, des facteurs comme: la susceptibilité de l'air, l'interférence, limitation de bande passante, l'allocation de fréquence, limitation d'énergie, les changements de cellules, etc. contribuent à la limitation des ressources affectant la qualité de service (QoS) ainsi que la sécurité.

L'article est organisé de la forme suivante: dans la section 2 est présenté un aperçu de quelques travaux similaires et l'architecture du *framework* développée, ainsi qu'une brève présentation du modèle de l'IETF/DMTF sont présentées dans la section 3. Les tests réalisés pour la validation de l'outil et ses résultats sont décrits dans la section 4. La section 5 présente la conclusion générale du travail et les possibles travaux futurs.

2. Travaux similaires

Plusieurs recherches ont été effectuées dans le domaine PBNM, principalement après le modèle standardisé par l'IETF/DMTF [2]. La nécessité, les expectatives et le futur de PBNM sont discutés par Shepard [9]. Les difficultés rencontrées dans la gestion de réseaux, les avantages et inconvénients de PBNM sont traités par Changkun [10]. Les exigences lors du développement de systèmes PBNM sont décrites par Mahon et al. [11].

Il existe aussi d'autres travaux qui cherchent à prouver les atouts de la Gestion de réseaux par Politiques développant des architectures et leur implémentation comme Ponnapan et al. [5] ou comme l'architecture QAME (*QoS-Aware Management Environment*) [12,13]. Hors du milieu

académique, Hewlett-Packard [14], ExtremeNetworks [15] et Cisco [16] ont aussi développé des outils pour la gestion par politiques des réseaux.

Malgré le nombre croissant de travaux dans le domaine, très peu d'outils developpés présentent les détails sur les fonctionalités et les performances, ce qui apporte des grandes difficultés dans la comparaison avec les autres travaux. Ceci a été un problème également rencontré par Ponnapan et al. [5].

3. Architecture LARCES_PBM

L'architecture normalisée par l'IETF/DMTF [2] est basée sur l'usage des politiques pour la gestion des réseaux. D'après Moore et al. [1], les politiques sont un ensemble de règles pour administrer, gérer et controler l'accès aux ressources des réseaux. Cette architecture est formée de quatre éléments basiques: un outil pour la gestion de politiques (*Policy Management Tool* – PMT), un repertoire de politiques (*Policy Repository* – PR), un point de décisions de politiques (*Policy Decision Point* – PDP) et un point d'application des politiques (*Policy Enforcement Point* – PEP) [2].

Le *framework* LARCES_PBM (Figure 1) possède tous les modules de l'architecture proposée par l'IETF/DMTF. Le PMT, le PDP et le PEP ont été développés en Java et par conséquent, sont multi-plateformes.

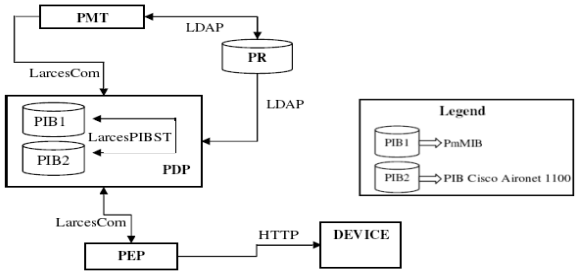


Figure 1. *Architecture du Framework LARCES_PBM*

Afin d'obtenir plus de scalabilité et renforcer le caractère générique, le *framework* est customisé à partir de fichiers XML. Altérant les paramètres de ces fichiers, il est possible de modifier le type de communication entre les modules, modifier la base de données utilisée, faciliter l'insertion des possibles modifications dans le modèle CIM *(Common Information Model)*, adapter le *framework* aux divers types de dispositifs. L'avantage de cet abordage est la non modification du code source. S'il survient une altération sur le modèle ou sur la forme de la politique appliquée, par exemple, il ne sera pas nécessaire d'effectuer la modification de l'implémentation pour que celle-ci s'adapte au nouveau traitement, ce qui conduit à une implémentation customisée.

3.1. Policy Management Tool (PMT)

Le PMT est une interface où l'administrateur du réseau cadastre les règles de politiques

(SLAs), qui sont traduites par le biais du modèle d'information utilisé, et ensuite *mapper* en un format compatible avec le schéma de stockage utilisé par le repertoire. Le modèle d'information utilisé a été le CIM[8], dû au fait qu'il décrit les informations importantes liées à la gestion dans un environnement réseau. Visant garantir que le SLA (contrat composé de conditions et des actions)soit appliqué, la politique doit passer par des processus de validation et de détection de possibles conflits avec les autres politiques déjà enregistrées. Au niveau du PMT également sont enregistrées les informations sur la topologie du réseau, les clients, les applications et les services qui vont être appliquées. Toute information insérée, enlevée ou modifiée au niveau du PMT est atualisée sur le PR. Le PMT, en plus de réaliser des fonctions extra comme réaliser la validation syntaxique, statique des données et la détection statique de conflits, est également responsable de présenter le contenu stocké dans le PR à travers un *browser*.

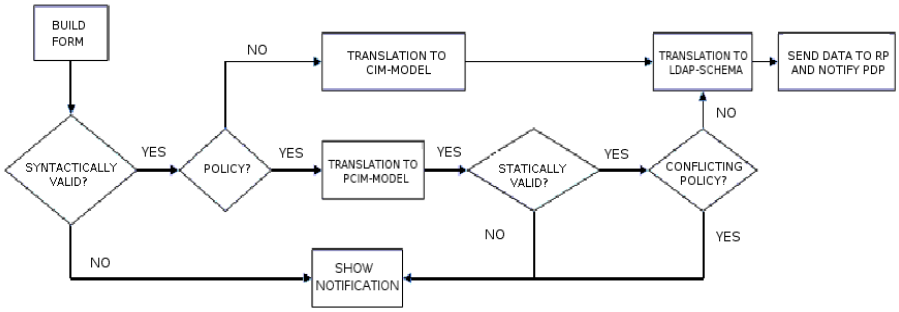


Figure 2. *Fluxogramme d'enregistrement des informations*

La séquence des actions exécutées dans l'enregistrement des informations est montrée dans la figure 2. Initialement est construit un formulaire en accord avec le type d'information inséré. Ensuite, est réalisée la validation syntaxique responsable de vérifier les données quant à la syntaxe avant que celles-ci soient traduites. Et le processus continue s'il ne se produit pas d'erreurs. Si la nouvelle information insérée est une politique, le formulaire est traduit dans le format PCIM (*Policy Core Information Model*) [1], sous modèle défini par CIM pour la représentation des politiques. La validation statique qui consiste à vérifier si les données sont en accord avec les capacités réseau est ensuite réalisée. Dans le cas qu'il ne se produit pas d'erreurs, l'étape suivante est la détection de conflits dont le rôle est de vérifier s'il existe une autre politique déjà enregistrée avec des paramètres conflictants. Il vient ensuite la traduction de PCIM dans le schéma de stockage LDAP, et les données sont alors enregistrées dans le repertoire. À la fin, une notification est envoyée au PDP. Dans le cas que l'information ne soit pas une politique, le formulaire est traduit dans le format de CIM_Network, autre sous modèle de CIM. Ensuite, est réalisée la traduction dans le schéma de LDAP, et les données sont alors enregistrées dans le repertoire. Une notification est également envoyée au PDP.

3.2 Policy Repository (PR)

Le PR utilisé dans le LARCES_PBM est responsable du stockage des informations des ressources du réseau, en plus des politiques. Il est le point central d'où sont tirées les informations de gestion de tout le réseau. Lors du développement du *framework,* il a été utilisé un serveur

LDAP. Le choix fut motivé non seulement par le fait qu'en plus d'être recommandé par l'IETF, nous avons la disponibilité d'un serveur de directoire libre, l'OpenLDAP qui a été utilisé dans ce *framework.*

Le modèle d'informations utilisé (CIM) est composé de divers sous modèles comme par exemple, celui du réseau (CIM_Network), et celui des politiques (PCIM). En ce qui concerne le PCIM, ont été utilisés deux RFCs, PCIM [1] et PCIMe [18]. Cependant, ce sous modèle n'est pas suffisant pour la description des politiques qui vise l'obtention de la QoS. Il a été ainsi nécessaire une extension de ces modèles pour le *framework.*

Pour que ces informations soient enregistrées, il est nécessaire que soit faite la traduction du modèle d'information dans le schéma de stockage du repertoire. Moore et al. [19] et Burnner et al. [20] définissent le *mapping* des sous modèles PCIM et PCIMe, respectivement, dans le schéma de stockage de LDAP. Ce *mapping* a été implémenté dans le *framework.* Il a été aussi réalisé le *mapping* des extensions.

3.3 Policy Decision Point (PDP)

Le PDP est le bloc qui peut être appelé le noyau de tout le *framework* et, pour cette raison, est plus complexe. Ce bloc est responsable de l'interprétation des politiques enregistrées dans le repertoire, prendre des décisions autour de ces politiques et ensuite les envoyer au PEP pour qu'elles puissent être appliquées. La séparation physique entre le PDP et le PEP a été faite considérant les avantages que ce choix apporterait sur l'aspect de l'agilité lors de la codification, la simplicité et principalement la scalabilité.

Les principales fonctions du PDP dans le LARCES_PBM sont: traduction de politiques, configuration des PIBs, détection dynamique de conflits, validation dynamique et distribution des politiques aux PEPs qui lui sont subordonnés, en plus de la découverte des ressources.

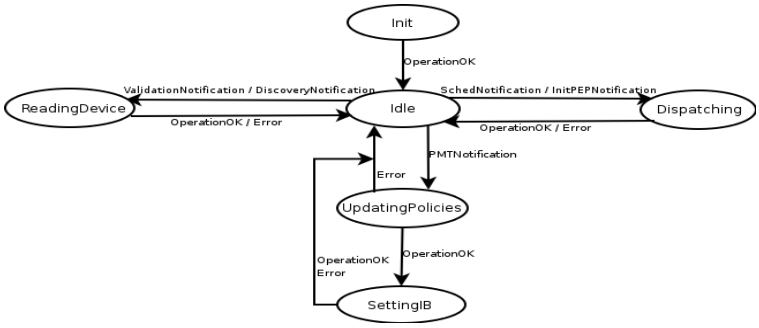


Figure 3. *Diagramme des états du PDP*

Le module PDP possède six états comme le montre le diagramme des états du PDP à la figure 3. La transition entre les états est réalisée à partir des événements.

Init est l'état d'initialisation du PDP. Dans cet état, sont obtenues en fichiers XML les configurations initiales, comme par exemple les protocoles à être utilisés, les adresses IP des

autres modules et les intervalles de temps pour les ordonnanceurs. L'autre déclenchement de *Init* est provoqué par un message de PMT au PDP pour que ce dernier obtienne les politiques présentes dans le PR. L'état *Idle* est responsable d'écouter et de traiter les différentes notifications. *UpdatingPolicies* est l´état responsable de l'actualisation des politiques. Les politiques traitées dans l'état *UpdatingPolicies* doivent être actualisées dans les PIBs et *SettingIB* est l'état qui réalise cette actualisation. *Dispatching* est l´état de distribution des politiques. C'est dans cet état que le PDP envoie au PEP les OIDs qui contiennent les informations des politiques qui doivent être appliquées.

La validation dynamique est réalisée au moment de la distribution de politiques. Lors de cette distribution de politiques au PEP, il est nécessaire de vérifier que les conditions actuelles du réseau permettent qu'elles soient appliquées. C'est seulement après ce processus, que l'on peut considérer que la politique est apte à être appliquée dans le système. La fonction de distribution est nécessaire pour que les politiques puissent arriver au PEP et qu'elles soient acheminées aux dispositifs où elles seront appliquées.

3.4 Policy Enforcement Point (PEP)

Étant donné les difficultés rencontrées dans les réseaux sans fils, comme étude de cas, le LARCES a développé un PEP s'appliquant à ce type de réseau, spécifiquement pour les Points d'accès du modèle *Cisco Aironet 1100*.

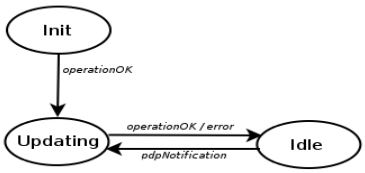


Figure 4. *Diagramme des états du PEP*

Le PEP possède trois états et trois types d'événements, comme illustrée dans la figure 4. L'état initial est *Init*. C'est l'état où sont faites toutes les configurations initiales, comme la communication avec les dispositifs gérés. Les paramètres nécessaires pour de telles configurations sont obtenus à partir d'un fichier XML. C'est dans l'état *Updating* que le PEP réellement applique les politiques aux dispositifs. L'état *Idle* est où le PEP simplement attend les nouveaux messages du PDP. *Idle* est l'état responsable de recevoir les notifications, vérifier l'émetteur, analiser et traiter le type de notification.

Dans le *framework,* chaque PEP possède un PIB crée pour enregistrer les données qui peuvent être manipulées dans les dispositifs qu'il gère. Le PIB créé pour ce PEP possède deux groupes basiques d'objets: *policies* e *access-categories*. Les actions liées aux politiques sont enregistrées dans le premier et le second contient les paramètres des classes de services présents dans le dispositif.

3.5 Communication entre les modules

Pour l'intégration entre les modules de l'architecture, il est évident la nécessité de communication entre eux. Les communications et les protocoles utilisés sont décrits dans le tableau 1.

Tableau 1. *Les modules et ses formes de communication*.

Modules	PMT-PR	PDP-PR	PMT-PDP	PDP-PEP	PDP-Dev	PEP-Dev
Communication	LDAP	LDAP	LarcesCom	LarcesCom	SNMP	HTTP

Toute la communication faite avec le PR est réalisée utilisant le protocole LDAP. Aussi bien le PMT que le PDP réalisent le stockage et la recherche sur le serveur du directoire (OpenLDAP) utilisant ce protocole. Cherchant une forme plus simple, rapide, d'implémentation facile, et qui s'adapte mieux aux nécessités du LARCES_PBM, le LARCES a développé une forme de communication, appelée LarcesCom (LARCES C*ommunication).* Celle ci a été la forme utilisée pour les communications entre le PMT et le PDP et entre le PDP et les PEPs. Cette forme de communication est basée sur l'envoi de messages simples par un module (émetteur) et l'envoi d'un message de confirmation de reception par l'autre module (récepteur). Dans le cas, que le message de confirmation n'est pas reçu, est effectué un re-envoi. Si une fois de plus, il n'ya pas de confirmation, le module récepteur est jugé inactif. Le format des messages du protocole est illustré dans la figure 5.

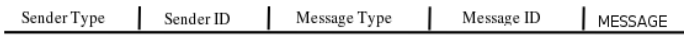


Figure 5*. Format des messages du protocole LarcesCom.*

Pour la communication entre le PDP et les dispositifs subordonnés au PEP, il est utilisé le protocole SNMP. L'objectif de cette communication est de monitorer l'état du réseau. Pour ceci est nécessaire la lecture de la MIB du dispositif à travers la commande *GET.* La communication entre les PEPs et les APs (Access Point) est faite avec le protocole HTTP, due à la facilité et la simplicité d'utilisation de ce protocole dans ce type de dispositif. Ainsi, les OIDs qui sont envoyés du PDP au PEP ont besoin d'être traduits par les URLs. La configuration est faite utilisant les commandes *GET* et *POST*, dépendant de l'URL.

4. Tests

Pour valider notre architecture, ont été réalisés des tests dans un réseau sans fils où ont été analisés les comportements des différents types de trafic avec et sans l'utilisation des politiques appliquées à ceux-ci.

4.1. Plate forme de tests

Pour l'exécution des tests a été utilisé un réseau (Figure 6), avec les technologies 802.11 et 802.3, constitué de cinq PCs *desktop*, un *switch,* un AP et deux PCs portables.

Les modules du LARCES_PBM ont été exécutés séparément sur quatre PCs *desktop*. Un autre PC *desktop* a été utilisé comme serveur (générateur de trafic) et les PCs portables ont été utilisés comme clients (récepteurs de trafic).

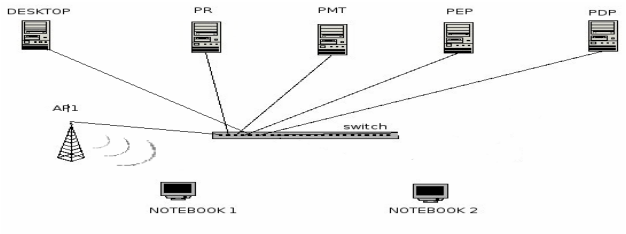


Figure 6. *Ambiance de tests*

Les PCs portables ont été connectés à travers le point d'accès *Cisco Aironet 1100.* Pour les interconnexions des PCs *desktop* et du point d'accès, a été utilisé un switch ENCORE ENH908-NWY+ de 8 ports. Les configurations utilisées sur les dispositifs sont décrits dans le tableau 2.

T*ableau 2.* *Configurations des dispositifs*

	Clients	**Server**	**PMT,PDP,PEP,PR**
CPU	Pentium IV 1600MHz	AMD Athlon XP 2800+	AMD Athlon XP 2800+
Memory	512MB	452MB	452MB
O.S.	Linux Kernel 2.6.12	Linux Kernel 2.6.12	Linux Kernel 2.6.8 / Windows XP
Iperf	Iperf Client	Iperf Server	N/A
Wireless card	WL110 11Mbps - COMPAQ	N/A	N/A
Ethernet card	N/A	EMB	EMB

4.2. Génération de trafic

Les mesures réelles dans les réseaux montrent que les trafics les plus fréquents sont HTTP, FTP, P2P, E-mail, vidéo et voix [22,23,24]. Dans le cas de ce travail, ont été choisis 4 types de trafic : HTTP, FTP, vidéo et voix.

Pour la génération des différents types de trafic, il a été utilisé le programme Iperf 2.0.2 qui permet les configurations comme la taille des paquets, le temps de génération du paquet, le MTU (*Maximum Transmit Unit*) et le type de protocole de transport (TCP/UDP). Les paramètres qui peuvent être mesurés à travers ce programme sont : le débit, la gigue, la perte de paquets pour les trafics UDP et pour TCP seulement le débit [25].

Pour tenter de s'approcher au maximum des modèles qui décrivent les comportements réels des trafics [26,27], ont été utilisés des scripts en *Perl* pour esssayer de parer aux insuffisances de Iperf. Par exemple, dans le cas de HTTP, le script a donné la possibilité de transférer diverses pages de tailles différentes et dans le cas du trafic vidéo a été effectuée la variation de la vitesse de transmission.

4.3. Configurations dans le *framework*

Dans le PMT ont été créés des SLAs qui utilisaient quatre services configurés : Or, Argent, Bronze et *Best-Effort.* La configuration donnant le meilleur service a été Or, suivi de Argent, Bronze et *Best-Effort* respectivement. Les valeurs des paramètres configurés dans les services et supportés par le point d'accès sont décrits dans le tableau 3.

Tableau 3*. Paramètres configurés dans les services*

Access Category	Min Contention Window (0-10)	Max Contention Window (0-10)	Fixed Slot Time (0-20)	Admission Control	Transmit Opportunity (0-65535 S)
Best Effort	5	10	7	NO	0
Bronze	5	10	3	NO	0
Prata	4	5	2	NO	6016
Ouro	3	4	2	NO	3264

4.4 Scénarios et résultats

Scénario 1: Variation de politiques avec le même trafic.

L'objectif du premier scénario est de vérifier si les politiques configurées dans le PMT vont été correctement appliquées dans le réseau. Pour cela, le trafic FTP a été généré à quatre moments différents, et appliquant à chaque fois un service différent.

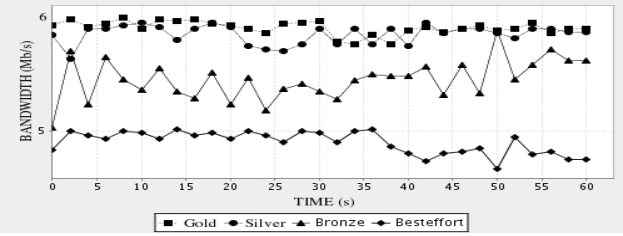


Figure 7. *Graphe bande(Mb/s) x temps(s) de FTP avec les 4 SLAs.*

La figure 7 illustre le graphe bande x temps résultant de ce scénario. Il est possible de s'apercevoir que plus le service utilisé est meilleur, meilleure est également la bande passante atteinte par l'application, en prenant en considération le fait que FTP est une application élastique. Avec le service Or, une bande passante de 6 Mb/s a été réussie, valeur atteinte à présent dans un réseau WLAN [28,29]. Par contre, avec le service *Best-Effort,* les 5 Mb/s n'ont pas été dépassés.

Ce résultat montre que le *framework* LARCES_PBM a appliqué correctement les politiques correspondantes aux SLAs utilisés.

Scénario 2: trafics simultanés avec et sans utilisation de politiques

Le second scénario permet une évaluation du comportement des quatres trafics avec l'application des politiques, et comparant les mêmes sans politiques.

Les quatre trafics différents (HTTP, FTP, vidéo et voix) ont été générés simultanément. Dans la première situation, il n'ya pas eu d'application de politiques, et dans la seconde, ont été appliquées les politiques Or pour le trafic voix, Argent pour le trafic vidéo, Bronze pour le trafic HTTP et *Best-Effort* pour le trafic FTP.

Afin de mieux illustrer l'influence de l'application des SLAs, ont été analysés les comportements de chaque type de trafic individuellement. Les graphes résultant de ces analyses sont illustrées ci-dessous. Dans le cas de la voix et la vidéo, sont illustrés les graphes de gigue et de perte des paquets (Figures 8, 9) où nous nous apercevons qu'avec l'utilisation des politiques, la variation de la gigue et les pertes sont très faibles. En ce qui concerne FTP et HTTP sont illsutrés les graphes de bande passante (Figure 10) où l'utilisation de la bande passante est meilleure avec l'application des politiques.

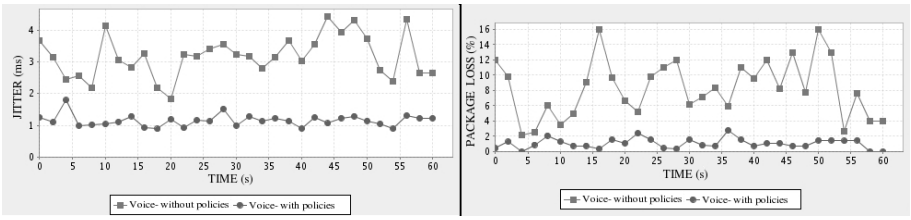


Figure 8 *Graphes de voix: à gauche, gigue(ms) x temps(s); à droite, perte de paquets(%) x temps(s).*

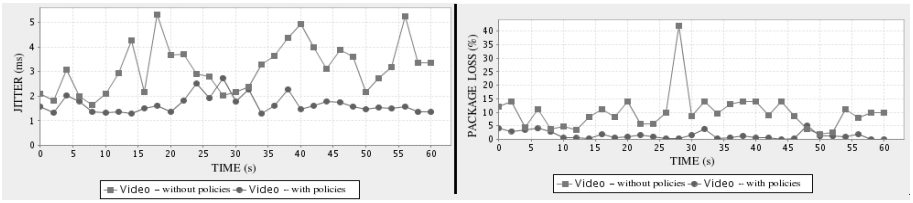


Figure 9. *Graphes de vidéo: à gauche, gigue(ms) x temps(s); à droite, perte de paquets(%) x temps(s)*

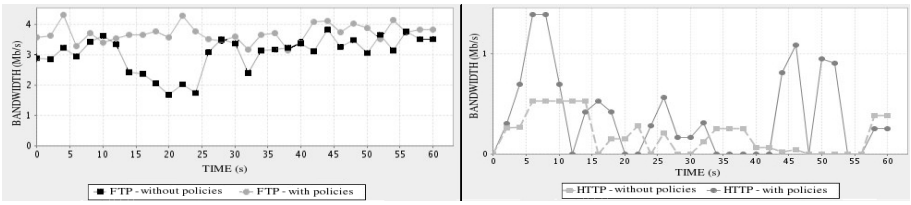


Figura 10. *Graphe bande(Mb/s) x temps(s): à gauche, FTP; à droite, HTTP***.**

Ces résultats démontrent bien qu'avec l'application des politiques, le comportement de tous les trafics a été sensiblement meilleur que le comportement des mêmes sans application de politiques.

Scénario 3: différents trafics avec et sans concurrence du milieu de transmission.

L'objectif de ce troisième scénario de tests (Figures 11, 12 et 13) est d'évaluer si l'application des politiques reste correcte lorsqu'il existe la concurrence du milieu de transmission. Dans ce scénario, le serveur est responsable d'envoyer vers deux clients (deux PCs portables) les quatre types de trafic avec politiques appliquées, mais cette fois FTP et le trafic vidéo à l'un et HTTP et le trafic voix à l'autre. Ensuite, nous ferons la comparaison avec le scénario précédent où tous les trafics étaient destinés à un unique client. La configuration des services appliqués à chacun des trafics est identique à celle utilisée dans le scénario précédent (scénario 2).

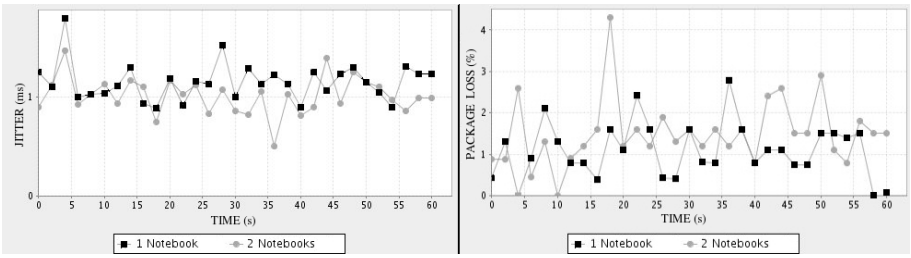


Figure 11. *Graphes de voix: gigue(ms) x temps(s), à gauche, et perte de paquets(%) x temps(s), à droite.*

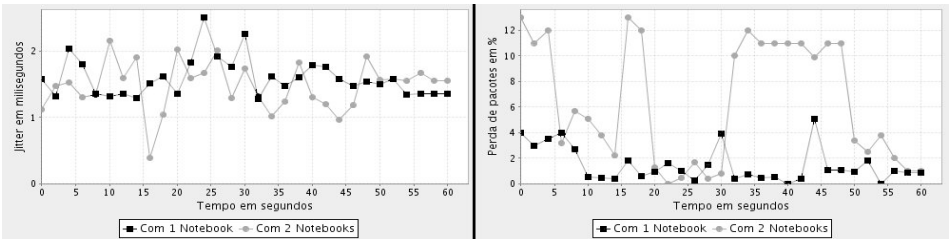


Figure 12. *Graphes de video: gigue(ms) x temps(s), à gauche et perte de paquets(%) x temps(s), à droite***.**

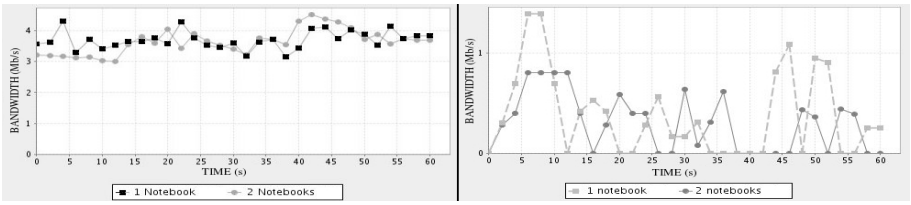


Figure 13. *Graphes bande(Mb/s) x temps(s): à gauche, FTP; à droite, HTTP*

Dans les figures, nous nous apercevons qu'il n'existe pas de grandes différences dans les 2 cas. Les figures 11 et 12 montrent les gigues et pertes de paquets, où il apparaît que les résultats sont légèrement meilleurs avec un PC portable. Dans le trafic HTTP (Figure 13) à deux moments isolés, la bande passante est largement différente avec un PC portable. À partir de ces résultats, il est possible de conclure que les configurations de politiques ont été également respectées avec deux PCs portables. Les petites différences obtenues avec les deux clients par rapport au cas d'un client témoignent bien de l'existence de la compétition du milieu de transmission.

5. Conclusion et travaux futurs

Dans cet article a été présentée et évaluée l'ímplémentation du *framework* générique LARCES_PBM pour la Gestion par Politiques de la QoS dans les réseaux de communication. Ce *framework* est applicable à tout type de réseaux, où il suffit à peine de l'adapter à chacun des types.

Les principales difficultés rencontrées dans le développement de ce travail ont été la réalisation de l'extension du modèle PCIM suivi du *mapping* de ce dernier par rapport au schéma de stockage du LDAP, le choix des protocoles de communication entre les modules et de la meilleure forme de manipulation des PIBs dans le PDP. Ainsi, l'un des apports de ce travail a été donc l'implémentation de l'extension et du *mapping* néccessaire à l'intégration des différents modules. Il a été également élaboré une nouvelle forme de manipulation des PIBs et une forme simplifiée de communication entre le PMT et le PDP et entre le PDP et le PEP.

Prenant en considération les difficultés rencontrées dans les réseaux sans fils, comme étude de cas a été développé un PEP spécifique pour la gestion de réseaux WLAN. La fonctionalité de l'architecture développée et de son implémentation a été démontrée à partir des tests réalisés ayant pour objectif l'obtention de la QoS dans ce type de réseaux. Les tests ont prouvé l'application correcte des politiques. Ils ont aussi démontré l'amélioration des comportements des trafics générés avec l'application des politiques. Il a été également démontré que les configurations des politiques sont aussi respectées lorsqu'il ya concurrence du milieu de transmission entre les PCs portables.

Les travaux futurs sont en relation avec l'implémentation d'un mécanisme pour la détection dynamique de conflits, au développement de PEPs pour les autres types de réseaux et également la réalisation des tests de performance du *framework*.

6. Bibliographie

[1] Moore, B., Ellesson, E., Strassner, J. e Westerinen, A. (2001) "Policy Core Information Model -- Version 1 Specification", Internet RFC 3060.

[2] Verma, D. (2000) "Policy-Based Networking: Architecture and Algorithms", New Riders, 1a edição.

[3] Westerinen, A., Schnizlein, J., Strassner, J. et al. (2001) "Terminology for Policy-Based Management", Internet RFC 3198.

[4] Yavatkar, R., Pendarakis, D. e Guerin, R. (2000) "A Framework for Policy-based Admission Control", Internet RFC 2753.

[5] Ponnapan, A., Yang, L. e Pillai, R. (2002) "A Policy Based QoS Management System for the IntServ/DiffServ Based Internet", Third International Workshop on Policies for Distributed Systems and Networks.

[6] Chan, K., Seligson, J., Durham, D. et al. (2001) "A COPS Usage for Policy Provisioning (COPS-PR)", Internet RFC 3084.

[7] Waldbusser, S. e Saperia, J. (2004) "Policy Based Management MIB", Internet draft draft-ietf-snmpconf-pm-15.

[8] "Common Information Model (CIM) Standards (2005)", http://www.dmtf.org/standards/cim/, Dezembro.

[9] Shepard, S. (2000) “Policy-Based Networks:Hype and Hope”, IT Professional, Volume 2 Issue 1, pages 12-16.

[10] Changkun, W. (2000) “Policy-based Network Management”, WCC – ICCT 2000.

[11] Mahon, H., Bernet, Y., Herzog, S. e Schnizlein, J (2000) “Requirements for a Policy ManagementSystem”, Internet draft draft-ietf-policy-req-02.

[12] Granville, L., Ceccon, M., Tarouco, L. et al. (2001) “An Approach for Integrated Management of Networks with Quality of Service Support Using QAME”, 12th International Workshop DSOM'2001, Nancy France.

[13] Granville, L. e Tarouco, L. (2001) “QAME – QoS-Aware Management Environment”, COMPSAC'01, p.269.

[14] Hewllett-Packard (2005) “HP OpenView PolicyXpert.”, http://www.managementsoftware.hp.com/products/pxpert/index.html, Dezembro.

[15] Extreme Networks (2005) “EPICenter Network Management Software”, http://www.extremenetworks.com/libraries/prodpdfs/products/epicenter.asp, Dezembro.

[16] Cisco Systems (2005) “Cisco QoS Policy Manager (QPM)”, http://www.cisco.com/warp/public/cc/pd/wr2k/qoppmn/, Dezembro.

[17] Barros, A., Fernandez, M., Campos, A. et al. (2005) “Implementação de validação de políticas e detecção de conflitos adaptáveis para uma arquitetura de gerenciamento baseada em políticas”, Hewllett-Packard, Relatório Técnico 06/2005.

[18] Moore, B. (2003) “Policy Core Information Model (PCIM) Extensions”, Internet RFC 3460.

[19] Strassner, J., Moore, B., Moats, R. e Ellesson, E. (2004) “ Policy Core Lightweight Directory Access Protocol (LDAP) Schema”, Internet RFC 3703.

[20] Burnner, M., Moron, D., Barba, A. e Reyes A. (2005) “Policy Core Extension Lightweight Directory Access Protocol Schema (PCELS)”, Ineternet RFC 4104.

[21] Aquino, F., Mendouga, L., Barros, A. et al. “Uma proposta para armazenamento de PIBs utilizando árvores”, Hewllett-Packard, Relatório Técnico 07/2005.

[22] Soule, A., Lakhina, A., Taft, N. et al. (2005). ”Traffic matrices: Balancing measurements, inference and modeling”, ACM SIGMETRICS’05, Banff, Canadá.

[23] Ziviani, A. e Duarto, O. (2005) “Metrologia na Internet”, Mini-Curso SBRC 2005.

[24] Metropolis (2005). “Metropolis: METROlogie Pour l'Internet et ses serices”, http://www.telecom.gouv.fr/rnrt/rnrt/projets/res_01_57.htm, Dezembro.

[25] Distribuited Applications Support Team (2005) “Iperf”, http://dast.nlanr.net/Projects/Iperf/, Dezembro.

[26] Choi, H.e Limb, J. (1999) “A Behavioral Model of Web Traffic”, ICNP'99.

[27] Anagnostou, M. et al. (1996) “A Mulitservice User Descriptive Traffic Source Model”, IEEE Transactions on Communications, vol 44.

[28] Schiller, J. (2003) “Mobile Communications”, Addison-Wesley, 2a edição.

[29] Rappaport , T. (2002) “Wireless Communications: Principles and Pratice”, Prentice Hall, 2a edição.

Validation de Politiques et Détection de Conflits pour la Gestion par Politiques de Réseaux: Une Implémentation customisée*

Ana Luiza Bessa de P. Barros, Joaquim Celestino Júnior, Laure Waha N. Mendouga, Marcial Porto Fernandez, André Luís de O. Campos, Fernanda Lígia R. Lopes, Francisco Wagner C. Aquino, Rafael Ramos Soares, Felipe Colares, Daniel Pereira

Departamento de Estatística e Computação – Universidade Estadual do Ceará (UECE)
Av. Paranjana,1700 – Campus do Itaperi – Fortaleza – Ceará - Brazil
{analuiza, celestino, laure, marcial, andre, fernanda, wagner,rafael} felipecolares, daniel@larces.uece.br

RESUMÉ: *PBNM est une forme de gestion susceptible à divers problèmes, comme par exemple, les politiques ne pouvant être appliquées pour manque de ressources ou la possibilité de désaccord, de conflits avec d'autres politiques. Ainsi, sont nécessaires des mécanismes qui valident les politiques et détectent les conflits entre elles. Ce travail présente une implémentation de ces mécanismes pour un outil de gestion basée sur l'architecture IETF/DMTF. Des schémas de validation syntaxique et sémantique, de forme statique et dynamique, ont été implémentés, ainsi que la détection statique de conflits. Le modèle CIM a été utilisé pour la manipulation des informations de politiques et des ressources réseaux. Il est également proposé une forme customisée afin de rendre de tels mécanismes adaptables à tout type d' altérations.*

ABSTRACT. *PBMN is a management technique susceptible to many problems, such as policies that cannot be applied because they are in disagreement with other policies or lack of resources. Thus, there is a need for mechanisms that validate policies and detect conflicts between them. This work presents an implementation of this mechanisms for a management tool based on IETF/DMTF architecture. It has been implemented models for syntactic and semantic validation, in static and dynamic ways, and static conflict detection. The CIM model was used to manipulate policies and network resources information. It is also proposed a form customized in order to make such mechanisms adaptable with any type of modifications.*

MOTS-CLÉS: *PBNM, validation, détection de conflits, customisée.*

KEYS-WORDS: *PBNM, validation, conflicts detection, customize.*

* Travaux realisés avec les ressources de HP Brasil P&D referents à la Loi nº 10.176 (Lei de Informática).

1. Introduction

Avec l'augmentation constante de la complexité des réseaux corporatifs, la nécessité de l'utilisation de modèles de gestion où l'on aura plus besoin de configurer séparement chacun des dispositifs est de plus en plus croissante. Une option viable a été le modèle de Gestion par Politiques de réseaux (PBNM - *Policy Based Network Management)*[1], lequel a pour principe d'abstraire l'administrateur des aspects de configuration du système et centraliser les informations de gestion de réseaux[1].

Les politiques sont constituées de règles qui définissent le comportement d'un système. Pour la représentation des politiques et des ressources du réseau, il est nécessaire l'utilisation d'un modèle d'information. Le DMTF (*Distributed Management Task Force*) fournit le CIM (*Common Information Model*) qui est un modèle de description des informations nécessaires dans un environement réseau [3]. Un autre élément nécessaire dans le PBNM est l'utilisation d'une architecture adéquate pour la traduction et la distribution des politiques dans le réseau. Pour ceci, l'IETF (*Internet Engineering Task Force*) et le DMTF proposent une architecture[1].

L'application d'une politique dans um système est un facteur critique, parce que nécessite la vérification des caractéristiques statiques et dynamiques. Cette vérification, appelée validation de politiques, est réalisée à partir du controle des paramètres à être configurés dans les dispositifs et de la vérification de la conformité du réseau au moment de l'application de la politique. En plus de la satisfaction de telles exigences, il est également nécessaire de s'assurer que chaque nouvelle politique ne présente pas de problèmes à coopérer ensemble avec celles déjà enregistrées, et cette propriété caractérise la détection de conflits.

Vu de telles nécessités, les modules de validation et de détection de conflits entre les politiques deviennent essentiels dans le processus de gestion de réseaux. Cependant, l'implémentation de ces mécanismes requiert l'identification de quel module (en terme de role) serait adéquat à la réception de chaque mécanisme (validation et détection) dans l'architecture normalisée par l'IETF/DMTF.

Dans cet article, sont proposées des adaptations dans l'architecture IETF/DMTF afin de donner un support aux mécanismes de validation des politiques et de détection des politiques, qui permettront la flexibilité dans la modification du modèle d'informations ou par exemple, de l'addition de nouvelles technologies à être gérées.

Dans les sections 2 et 3 sont exposés les concepts de validation et de détection de conflits, respectivement. Les adaptations proposées afin de customiser l'architecture sont décrites dans la section 4, pendant que dans la section 5 l'implémentation réalisée est décrite dans toute sa structure. La section 6 présente la conclusion et les travaux futurs.

2. Validation de politiques

Dans un scénario réel de PBNM, de multiples politiques seront appliquées aux éléments de réseau pour satisfaire aux requêtes de diverses applications et des utilisateurs dans différents

domaines de configuration. Ainsi, la politique passe par diverses étapes, depuis son enregistrement au niveau du PMT jusqu´à ce qu'elle soit effective comme configuration sur le dispositif du réseau. Cependant, durant ce processus peuvent se produire par exemple, des données inconsistentes ou des politiques pouvant être en dehors des configurations acceptables par le dispositif. Un autre problème possible est la non disponibilité d'une ressource gérée au moment de l'application de la politique.

La validation de politiques a pour role de vérifier si une politique est apte à servir comme ensemble de règles qui définissent le comportement d'un système et de s'assurer de son application au niveau des dispositifs gérés. Le processus de validation doit incorporer aussi bien des vérifications syntaxiques que sémantiques [1]. Toute fois en [2], sont considérés des aspects de contrôle au moment de l'enregistrement et de l'éxécution de la politique, ce qui subdivise le processus de validation en controle statique et dynamique. Dans ce travail est présentée une approche qui combine les propositions de [1] et [2] pour les mécanismes de validation.

2.1. Validation syntaxique

La validation syntaxique a pour fonction d'assurer que la syntaxe d'une politique est correcte. La non réalisation d'une vérification syntaxique dans les politiques enregistrées peut apporter des problèmes au moment de l'application et de l'exécution de ces dernières, parce qu´il peut se produire des inconsistances dans les informations passées au système de gestion. Dans l'exemple à suivre est démontrée une pseudo-définition de politique, où l'on peut vérifier une donnée syntaxiquement invalide. L'adresse IP du destinataire n'est pas dans les limites déterminées pour un champ IP.

```
target  192.168.1.257
if      (source = 192.168.1.7 &&
        destination=192.168.1.4);
apply service=GOLD to packages;
```

2.2. Validation sémantique

Considérant la validation sémantique comme responsable du controle de l'adéquation des politiques par rapport aux caractéristiques et aux états du réseau, peuvent être observés deux types de vérification : statique et dynamique.

La validation statique est utilisée pour la vérification de la possibilité qu'une règle soit appliquée au système géré. Ce controle est réalisé au moment d'enregistrer la politique[2]. Dépendant de la politique insérée, il pourra être vérifié si le dispositif géré supporte la fonctionalité introduite dans la requête ou si la configuration à être réalisée n'excède pas les limites des paramètres à être modifiés. Par exemple, une politique qui envoie la requête d'une bande passante supérieure à celle supportée par un lien de communication devra être considérée invalide.

Le controle dynamique est réalisé au moment de l'application de la politique et est responsable de vérifier si une ressource est disponible pour satisfaire une requête demandée.

3. Détection de conflits entre politiques

Il n'est pas suffisant de s'assurer que la politique sera correctement appliquée au réseau grâce au processus de validation, parce que celle-ci peut présenter des problèmes au moment de l'appliquer par rapport à l'ensemble des politiques déjá insérées. Nous rappelons que les politiques sont spécifiées par un ensemble de règles avec la sémantique "condition" donc "action". Un ensemble de politiques est dite consistente lorsqu'il ne peut être trouvé des contradictions entre elles, autrement dit deux (ou plus) politiques appliquées sur certaines conditions, peuvent avoir leurs objectifs ("actions") simultanément satisfaites. Dans le cas contraire, cela se caractérise par un conflit[1].

P1: Tout le trafic Vidéo doit recevoir le service GOLD.

P2: Tout utilisateur du département Réseau doit recevoir le service SILVER.

Supposons que les deux politiques soient correctes et valides. Cependant il est facile de noter qu'elles présenteront des problèmes à agir simultanément. Dans le cas qu'un utilisateur du département de Réseau utilise une application de vidéo, comment déterminer le service qui lui sera appliqué GOLD ou SILVER.

La finalité fondamentale de la détection de conflits est analyser les spécifications de la politique afin de fournir un profil des types de conflits qui se produisent dans le systéme. Selon[4], il ya deux catégories de conflits qui nécessitent être traités indépendament: conflits statiques et conflits dynamiques.

Le conflits statiques sont ceux dont les caractéristiques peuvent conduire à des inconsistences de l'application de la politique. Ils ne prennent pas en compte la situation actuelle du réseau, par conséquent ces caractéristiques doivent être détectées au moment de l'insertion de la politique dans le système. Les conflits dynamiques à son tour, ont la charge de l'imprévisibilité appliquée à elle, dû à sa dépendance dynamique aux facteurs comme les évènements ou caractéristiques mutables. Ce type de conflits fait qu'il est nécessaire la monitoration constante des conditions afin que la détection soit bien suivie.

4. Adaptations sur l'Architecture de l'IETF/DMTF

L'IETF/DMTF propose une architecture pour le PBNM (Figure 1), laquelle est constituée de quatre modules : un outil pour la gestion de politiques (*Policy Management Tool* – PMT); un repertoire de politiques (*Policy Repository* – PR); un point de décisions de politiques (*Policy Decision Point* – PDP) et un point d'applications de politiques (*Policy Decision Point* – PEP) [1]. Bien que soient définies des caractéristiques des modules, certains détails d'implémentation ne sont pas normalisés comme par exemple, les protocoles de communication ou quels sont les blocs responsables de la réalisation de la validation et de la détection de conflits.

Le PMT est le module utilisé pour l'insertion des politiques dans un langage de haut niveau, de forme à abstraire l'administrateur des aspects de configuration de chaque dispositif. Il est également responsable de la traduction de telles politiques à un niveau plus détaillé de configurations et de leur enregistrement dans le PR.

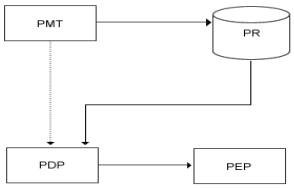


Figure 1: *Architecture proposée par l'IETF/DMTF*

Cette base de données enregistre les informations de gestion du réseau, comme par exemple, les politiques, les dispositifs, les clients et applications du réseau. Après l'envoi d'une politique au PR, le PMT notifique le PDP, et ce dernier réalise toutes les vérifications logiques pour savoir à quel moment cette politique doit être distribuée au système. Le PDP est également responsable de la traduction des politiques à une forme compréhensible par les PEPs, lesquels iront finalement réaliser les configurations des dispositifs nécessaires pour l'application de la politique dans le système.

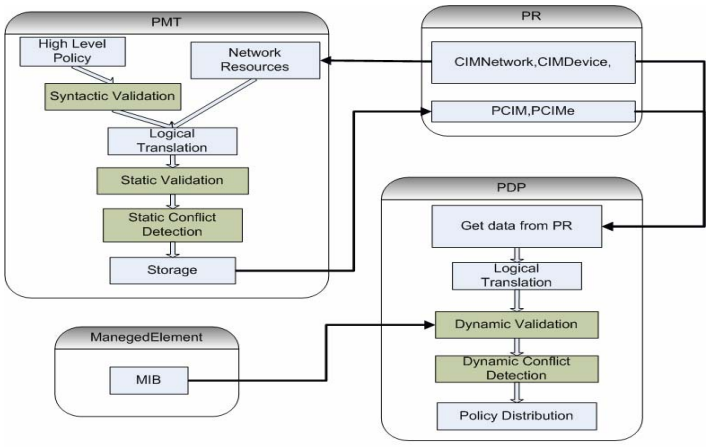


Figure 2*: Modifications proposées dans ce travail.*

Il est proposé en [1] que aussi bien le processus de validation syntaxique que celui de la validation sémantique restent dans le PMT dans un module de transformation logique ("*Policy Transformation Logic Module*"), lequel aussi serait responsable de la détection des conflits. Toute fois, dans un cas d'implémentation réel [5,6], il a été observé la nécessité d'impliquer le PDP dans

ces processus, simplement parce que le PMT ne serait pas adéquat aux vérifications des caractéristiques dynamiques du réseau.

Ce travail fait une proposition de modifications de l'architecture définie par l'IETF/DMTF conforme montrée dans la figure 2, dans les blocs PMT et PDP.

4.1 Validation et Détection de conflits au niveau de PMT

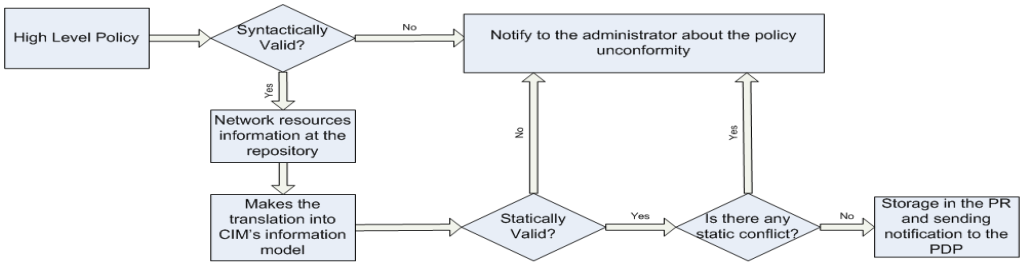


Figure 3: *Fluxogrammme de fonctionement du PMT***.**

Le PMT est responsable de fournir une interface pour l'insertion des politiques. Il peut être installé dans une ou plusieurs machines du réseau. L'autre caractéristique importante est que le module n'a pas besoin d'être actif durant le processus de gestion, c'est-à-dire il peut être désactivé après l'insertion des politiques. Dans la figure 3, est présenté le fluxogramme de fonctionement du PMT.

La validation syntaxique est réalisée après insertion de la politique dans un langage de haut niveau, au niveau de l'interface graphique. Cette information pour l'instant n'a pas besoin d'être traduite par le modèle PCIM pour être validée syntaxiquement, parce que la vérification dépend à peine de la forme de la spécification de la politique[1]. Basé sur cet aspect, la validation syntaxique est donc réalisée avant le processus de traduction afin d'éviter la réalisation des taches non nécessaires dans le cas d'insertion de politiques invalides.

Ce processus terminé, le PMT obtient du PR les informations sur la structure du réseau (CIM Network) pour la réalisation de la traduction de la politique utilisant PCIM. Ensuite sont réalisées les vérifications des informations statiques du réseau. De telles données décrivent les caractéristiques fixes de configurations des dispositifs et sont disponibles au moment de l'insertion de la politique. Et c'est ainsi que le PMT semble le meilleur candidat pour la réalisation de ces mécanismes qui sont la validation statique et la détection de conflits statiques.

La détection de conflits statiques est réalisée juste après la validation statique, comparant les paramètres de la politique à être appliquée avec ceux des politiques déjà enregistrées dans le PR. De cette forme, l'étape de détection statique dans le PMT visa à ménager tout le système dans le cas de politiques conflictantes.

4.2 Validation et détection de conflits au niveau du PDP

Le PDP est le bloc qui peut s'appeler le noyau de tout le *framework,* et pour cette raison le plus complexe. Au contraire du PMT, le PDP doit être toujours actif durant tout le processus de gestion, parce que c'est lui qui vérifie l'adéquation des conditions pour lesquelles une politique doit être appliquée en tenant compte de la situation actuelle du réseau. De telles fonctionnalités font que le PDP est le plus adéquat à recevoir les validations qui engagent des informations dynamiques du réseau. Les mécanismes implantés dans le PDP, conforme la figure 2, sont décrits comme suit :

Pour la réalisation de la validation dynamique, il est nécessaire que le PDP traite les informations actuelles (courantes) du réseau, lesquelles peuvent être obtenues à partir du PR ou en les cherchant au niveau des dispositifs.

L'implémentation de la détection dynamique doit être abordée seulement dans les travaux futurs.

5. Implémentation des mécanismes

Pour l'implémentation des mécanismes de validation et de détection de conflits proposé dans cet article, il est considéré la possibilité d'expansion des ressources gérées et la constante modification du modéle d'informations CIM, parce que ce sont des facteurs qui provoquent des modifications brusques dans ces mécanismes.

L'introduction d'un nouveau dispositif dans l'architecture requiert que les mécanismes de validation et de détection soient altérés, parce que créent de nouvelles conditions de gestion. Un exemple serait l'addition d'un routeur avec la technologie IntServ, par exemple. De tel dispositif supporterait une série de caractéristiques de gestion avec de nouvelles formes de définition de politiques. Dans ce cas, pour que la validation de politiques et la détection de conflits fonctionnent, il va falloir diverses altérations sur l'implémentation de ces mécanismes afin de tenir compte de ces nouvelles caractéristiques.

L'autre facteur qui peut impliquer la réimplémentation de ces mécanismes est la modification du modèle d'information utilisé. Cette modification peut être causée par deux facteurs : la nécessité de l'extension du modèle afin de supporter de nouvelles ressources ou de simples améliorations.

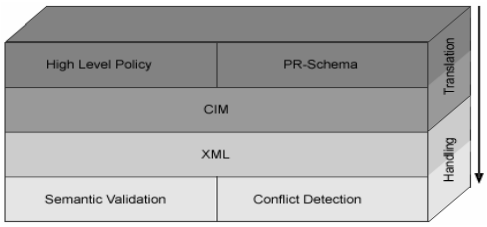


Figure 4: *Architecture pour la traduction et validation sur CIM.*

La couche XML permet des altérations du modèle d'information, aussi bien l'addition de nouveaux dispositifs sans la nécessité des modifications sur les mécanismes de validation et de détection de conflits. Par exemple, s'il se produit une modification du modèle CIM comme une classe qui n'existe plus ou remplacée, il ne sera pas nécessaire d'altérer tous les mécanismes de validation et détection qui utilisent de telles données. Il serait suffisant de modifier le fichier XML qui indique les informations (classes et attributs) utilisées dans le modèle. Un autre cas serait l'addition d'une nouvelle technologie à être gérée. Pour cela, il suffira à partir de ce fichier XML, incluire les nouvelles informations utiles dans les processus de validation et de détection. Il a été choisi l'utilisation de XML dû à l'ample acceptation de cette norme pour la spécification de configuration de systèmes.

Les politiques et informations de ressources réseaux peuvent être obtenues dans deux situations distinctes: Entrée de politiques dans un langage de haut niveau au PMT ou récupération des données déjà enregistrées au PR, ce dernier est réalisé par le PDP. Subséquement, les données sont traduites utilisant le modèle CIM. Après lorsqu'il sera nécessaire de réaliser la validation ou la détection de conflits, seront cherchés dans les fichiers XML quelles sont les données du modèle CIM à être utilisées et traitées dans ces processus.

La figure 5 ci-dessous montre un exemple d'une partie d'un fichier XML utilisé pour configurer les mécanismes de validation statique et dynamique dans le *framework* LARCES PBM [5,6]. Dans ce fichier sont spécifiés deux attributs qui devront être validés pour les politiques de QoS appliquées sur un AP (Access Point) de Cisco, lequel est géré par le prototype du *framework.* Ce fichier, par exemple, définit que l'attribut « MaxWindowContention » de la classe « APCiscoService » doit être vérifié.

Cette classe fut étendue à partir du modéle d'information afin de subvenir aux nécessités de gestion de ce *framework.* Cet attribut sera vérifié dans la validation statique et sa valeur sera obtenue du PR à partir de l'instanciation dynamique pour le traitement de la classe et des données. Étant donné qu'il n'est pas spécifié le type de comparaison ou les valeurs non valides, le système réalisera une comparaison de limite entre cette valeur et la valeur à être configurée dans la politique.

```
- <Services>
  - <Service name="Gold">
    - <UsedAttributes>
        <attribute name="MaxWindowContention" dinamicValidation="false" staticValidation="true" class="APCiscoService"
          AccessDevice="false" UpdateLDAP="false" isPIBAccess="true" defaultIndex="1" />
      - <attribute name="PowerState" dinamicValidation="true" staticValidation="false" class="UnitaryComputerSystem"
          AccessDevice="true" UpdateLDAP="true" isPIBAccess="false" defaultIndex="1">
          <InvalidValue index="1">6</InvalidValue>
          <InvalidValue index="2">7</InvalidValue>
          <InvalidValue index="3">8</InvalidValue>
          <InvalidValue index="4">9</InvalidValue>
        </attribute>
      </UsedAttributes>
    </Service>
  </Services>
```

Figure5: *Configurations des paramètres de Validation em XML.*

5.1 Implémentation de la validation statique

L'algorithme 1 décrit les étapes réalisées dans le mécanisme implémenté dans le framework. Initialement, la classe de service dont appartient la politique est obtenue et le dispositif à être configuré. En accord avec ces informations, les paramètres (“attributs”) que seront vérifiés durant la validation sont enregistrés en XML. Dans cette étape seront vérifiés à peine les attributs indiqués pour la validation statique.

Algorithm 1: Static Validation

```
Input: policy
service ← getProvidedService(policy);
device ← getDevice(policy);
attributes ← getAttributes(service, device);
foreach attributes[] do
    attributeValue ← getAttributeValueAtPolicy(attributes[i], policy);
    deviceValue ← getValueAtPR(attributes[i]);
    if isAttributeInvalid(attribute, attributeValue, deviceValue) then
        error( “Invalid Attribute: ”, attributes[i]);
    end
end
```

Pour chaque attribut, la valeur qui doit être configurée utilisant les informations contenues dans la politique à être appliquée, est représentée par « AttributeValue ». Par conséquent, les informations de restriction pour cet attribut seront cherchées dans le PR, et enregistrées dans « Device Value». Finalement, la méthode « attributeInvalid() » vérifie si la valeur de l'attribut (« AttributeValue ») est en accord avec les restrictions (« Device Value») selon la vérification qui doit être réalisée sur l'attribut (« attribute[i]).

5.2 Implémentation de la détection de conflits statiques

L'implémentation que nous présentons est celle adaptée pour le framework de gestion LARCES_PBM. L'algorithme 2 illustre l'implémentation de ce mécanisme.

Initialement, est obtenu le dispositif configuré par la politique en question. La finalité est cherchée dans le PR celles-là qui également seront appliquées sur le dispositif et les enregistrées dans devicePolicies. Ensuite pour chaque politique présente dans « devicePolicies », est faite la comparaison de ses champs avec ceux de la politique « policy » à être insérée.

La première condition comparée est celle du temps. La méthode “compareTimes()” vérifie s'il ya un intervalle de temps qui coincide entre la période de validité des deux politiques. Dans le cas que le retour est vrai, l'analyse se poursuit afin de comparer les autres conditions. Si c'est « faux », il n'ya pas de possibilité de conflits entre les politiques analisées, déjà que les deux n'agiront jamais simultanément.

Due à la coincidence dans l'intervalle de temps, la méthode “compareConditionsAttributes()” cherche les prochaines conditions dans le fichier XML et obtient les attributs qui seront comparés en relation à chaque condition.

Algorithm 2: Static Conflict Detection

```
Input: policy
device ← getConfiguredDevice(policy);
PRPolicies ← getPoliciesAtPR();
foreach PRPolicies[] do
    PRDevice ← getConfiguredDevice(PRPolicies[i]);
    x ← 0;
    if device = PRDevice then
        devicePolicies[x] ← politicasPR[i];
        x++;
    end
end
foreach devicePolicies[] do
    if compareTimes(policy, devicePolicies[i]) then
        if compareConditionsAttributes(policy, devicePolicies[i]) then
            if compareActionsAttributes(policy, PRPolicies[i]) then
                error(“Coincidence Policies”);
            else
                error(“Conflict Detected”);
            end
        end
    end
end
savePolicy(policy);
```

S’il se produit des coincidences de toutes les valeurs des attributs, alors suit la méthode “compareActionsAttributes()” qui obtient en XML les actions à être configurées. Si celles-ci s'avérent identiques, alors nous avons des politiques redondantes. Dans le contraire, il y aura conflit. Dans ces situations, l'administrateur est avisé que la politique à être enregistrée est redondante ou qu'il existe un conflit.

5.3 Implémentation de la validation dynamique

L'algorithme 3 montre les détails des étapes du processus de la validation dynamique.

De la même forme que la validation statique, dans la validation dynamique sont obtenues les informations de type service, du dispositif à être géré, et les attributs à controler. Aussi sera obtenue la valeur qui doit être configurée utilisant les informations contenues dans la politique à être appliquée, et cette valeur restera enregistrée dans "AttributeValue". Cependant, dépendant de l'attribut les informations sur le dispositif pourront être obtenues à partir du PR ou accessant le dispositif directement. Mais la forme d'obtention sera indiquée en XML, dépendant de l'attribut. L'information collectée sur la situation actuelle du réseau restera collectée dans la variable "deviceValue".

La méthode "invalidAttribute()" est responsable de réaliser la comparaison nécessaire entre la valeur à être configurée (“AttributeValue”) et la valeur du dispositif (“deviceValue”). La forme de vérification qui sera réalisée peut être rencontrée dans XML en accord avec l'attribut (“attribute”)

passé comme paramètre. Dans le cas que l'attribut soit invalide (la méthode "invalidAttribute" retournera "vrai"), et ce dernier sera additionné dans la liste des attributs invalides.

```
Algorithm 3: Dynamic Validation
  Input: policy
  service ← getProvidedService(policy);
  device ← getDevice(policy);
  attributes ← getAttributes(service, device);
  foreach attributes do
  |  attributeValue ← getAttributeValueAtPolicy(attributes[i], policy);
  |  if isAccessDevice(atributos[i]) then
  |  |  deviceValue ← getValueAtDevice(attributes[i], device);
  |  else
  |  |  deviceValue ← getValueAtPR(attributes[i]);
  |  end
  |  if isInvalidAttribute(attribute, attributeValue, deviceValue) then
  |  |  addInvalidAttribute(attributes[i]);
  |  end
  end
  if existsInvalidAttribute() then
  |  action ← getAction(getInvalidsAttributes());
  |  executAction(action);
  end
```

Après avoir vérifié tous les attributs qui seront configurés dans le dispositif, s'il yavait un attribut invalide, une action doit être prise par le f*ramework*. Toutes les actions restent enregistrées en XML, cependant il doit être choisi l'action appropriée pour tous les problèmes rencontrés durant le processus de validation. Ce choix est fait à travers la méthode "executAction()", lequel ira vérifier la meilleure action.

6. Tests

L'implémentation proposée pour les mécanismes de validation et de détection de conflits a été réalisée sur l'outil LARCES_PBM. Le résultat de la validation syntaxique et statique effectué est affiché comme le montrent les figures 7, 8 et 9. Ceci a été réalisé lors de l'enregistrement d'une classe de service pour la gestion de l'AP (Acces Point) de cisco 1100.

Figure 6*: Enregistrement invalide des informations de classe de services.*

Figure 7: *Validation syntaxique*

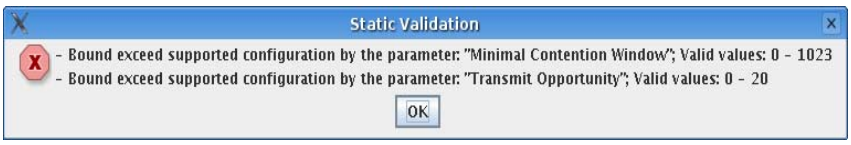


Figure 8: *Validation sémantique statique*

L'enregistrement exposé dans la figure 6 a produit les notifications d'invalidité tant syntaxique que sémantique des données insérées. La première vérification réalisée a causé le lancement de la notification de service invalide syntaxiquement (figure 7), parce qu'un champ numérique a été configuré par de la litérature. La seconde notification (figure 8) s'est produit à cause des paramètres de « Minimal Contention Window » et « Transmit Opportunity » qui ont dépassé les limites de configuration supportées par le dispositif.

La validation dynamique a été également réalisée. Dans le cas de l'implémentation du framework est vérifié si le dispositif à être configuré (AP cisco) est actif au moment de l'application de la politique. La figure 9 montre le processus de validation dynamique réalisé sur le PDP avant l'application de la politique.

```
debian:~/Desktop/FrameWork4# ./PDP.sh
Initializing PDP...
Thread para recepção de mensagens startado
Thread para recepção de mensagens startado
Iniciado Timer para Descorta de Recursos: 40000
/root/Desktop/FrameWork4/pastaPDP/config/ConnectorConfig.xml
--------------------------------------------------------------------------------
End of PDP cycle...State = IDLE
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Received policy by PMTNotification = PMT#PMT#A#RegraTeste_At_192.168.1.151
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Updating Policies
--------------------------------------------------------------------------------
Startind Dinamic Validation on policy RegraTeste_At_192.168.1.151
--------------------------------------------------------------------------------
Invalid policy:
  -->Device state is: shut down
--------------------------------------------------------------------------------
Sending error message to PMT:
PDP#pdp1#cn=IPAddress_200.129.22.70,cn=RegraTeste_At_192.168.1.151,ou=Rules,
ou=pbm,dc=larces,dc=uece,dc=br#pcelsVariableDN#
--------------------------------------------------------------------------------
estabelecendo socket...
--------------------------------------------------------------------------------
End of PDP cycle...State = IDLE
--------------------------------------------------------------------------------
```

Figura 9 – *Validation dynamique*

La figure 9 montre le debug du PDP. Il se produit une vérification si le dispositif à être configuré (192.168.1.151) n'est pas actif. Ainsi une notification est envoyée au PMT informant le problème à l'administrateur.

7. Conclusion et travaux futurs.

Dans ce travail, est proposée une forme d'implémention de la validation de politiques et la détection de conflits statiques basée sur la standardisation de l'IETF/DMTF.

Il a été pris comme base les propositions de [1] et [2] conforme décrite dans la littérature. Dans [1] la validation syntaxique et sémantique sont au niveau du PMT, lequel ne prend pas en considération l'influence des aspects dynamiques dans l'application des politiques.

SLOMAN propose un mécanisme de validation utilisant le PONDER comme langage de spécification de politiques et les comparant avec les informations contenues dans CIM.

Dans ce travail est proposé que la validation et la détection de conflits de politques soient réalisées sur le modéle d'informations sans prendre en compte le langage de spécification, à travers la proposition d'une architecture qui tourne toute l'implémentaion adaptable, autrement dit une implémentation customisée.

Comme continuation de ce travail, il est suggéré l'amélioration des propositions présentées, en plus du développement d'un modèle pour la détection dynamique de conflits. Il est pertinent de citer aussi la résolution de conflits comme une autre ligne de recherches à être considérée afin de rendre le système plus complet.

8. Bibliographie

[1] Verma, D. C, "Policy-Based Networking – Architecture and algorithms", New Riders, Novembro de 2000.

[2] Sloman, M., Lymberopoulos L., Lupu, E., "PONDER Policy Implementation and Validation in a CIM and Differentiated Services Framework", Maio de 2004.

[3] Tutorial sobre o CIM fornecido pelo DMTF. Página da WWW (World Wide Web) url: http://www.dmtf.org/education/tutorials, acessado em 22/12/2005.

[4] Dunlop, N., Indulska, J., Raymond, K., "Dinamic Conflict Detection in Policy-Based Management Systems", 2002.

[5] Bessa, Ana Luiza de P. B., Fernandez, M. P., Celestino, J. J., et. Al, "Un Outil Générique de Gestion par Politiques: Validation de l'Implémentation dans un réseau sans fils WLAN" GRES 2006.

[6] Westerinen, A., Schnizlein, J., Strassner, J. et al. (2001) "Terminology for Policy-Based Management", Internet RFC 3198.

[7] Moore, B., Ellesson, E., Strassner, J., "Policy Core Information Model", Request for Comments 3060, Fevereiro de 2001.

[8] Moore, B., "Policy Core Information Model (PCIM) Extensions", Request For Comments 3460, Janeiro de 2003.

Aspects of Configuration Constraints in Network Services

Rudy Deca [*] — **Omar Cherkaoui** [*] — **Yvon Savaria** [**] — **Doug Slone** [***]

[*] *Université du Québec à Montréal (UQÀM),*
Pavillon Sherbrooke SH-5715, 200 rue Sherbrooke O.,
Montréal, QC H2X 3P2, Canada.

deca@info.uqam.ca, cherkaoui,omar@uqam.ca

[**] *École Polytechnique de Montréal,*
Pavillon Mackay-Lassonde M-5117,
2700 Chemin de Polytechnique,
Montréal, QC H3T-1J4, Canada.

yvon.savaria@polymtl.ca

[***] *Cisco Systems Canada Co.,*
3000 Innovation Dr., Kanata, ON K2K 3E8, Canada.

dslone@cisco.com

ABSTRACT. *As Internet grows, the services provided over it become more complex and therefore more difficult to manage. New issues related to the service deployment, upgrade and repair arise constantly and require adequate solutions. Some of these issues are caused by various types of configuration constraints, which are presented, analyzed and classified in this paper. We argue that a good analysis of configuration constraints will facilitate the quest for adequate solutions to these issues.*

RÉSUMÉ. *Au fur et à mesure que l'Internet s'agrandit, les services réseau y fournis deviennent plus nombreux et, en conséquence, plus difficiles à gérer. De nouveaux défis reliés aux déploiement, réconfiguration et rétablissement des services, apparaissent constamment et demandent des solutions adéquates. Certains de ces défis sont causés par différents types de contraintes de configuration, qui sont présentés, analysés, et classifiés dans cet article. Nous sommes d'avis qu'une bonne analyse des contraintes de configuration facilitera la recherche des solutions adéquates à ces défis.*

KEYWORDS: *network service, network service configuration, configuration operation, configuration constraint, configuration dependency, validation rule.*

MOTS-CLÉS : *service réseau, configuration des services réseau, opération de configuration, contrainte de configuration, dépendance de configuration, règle de validation.*

1. Introduction

As the Internet constantly develops and provides an increasing number of services and features to the customers, the difficulty of the network service configuration tasks, increases accordingly. For instance, multiple equipments of a network may have traffic filters and IPsec services. Multiple filters may also be configured on the same equipment. Each traffic filter consists of tens or hundreds of parameters. All the filters on an equipment and on different equipments must be compatible. Since the equipments are provided by different vendors, the traffic filters are different. When the customer changes its requests, the traffic filters must be changed to match the new demands.

Several main management approaches have been proposed and used for the resolution of network management problems. The most widespread configuration approach, accounting for around 90 % of the network management, uses native user interfaces provided by the operating systems of the equipments, such as the Command-Line Interfaces (CLIs) (e.g. Cisco's IOS and CatOS – for earlier switches, Juniper's JUNOS, Lucent's ComOS, Nortel's TI, etc.) or GUIs (Nortel's Configuration Manager, Lucent's PMVision, etc.). Other important management approaches are: (1) the Simple Network management Protocol (SNMP) proposed by the IETF for Internet management (IETF RFC1359, etc.), but used mainly for monitoring; (2) the Opens Systems Interconnection (OSI) System Management, (proposed by the ISO) which defines a Reference Model, a meta-language for the description of the managed objects, the Guidelines for the definition of managed objects (GDMO) (ISO/IEC, 1991), and a management protocol, the Common Management Information Protocol (CMIP) (IETF RFC1098 and RFC1189), etc. The CMIP is used by the Telecommunications Network Management (TMN) generic framework proposed by ITU-T (ITU-T, 1992) for the management of telecommunications devices; (3) Web-Based Enterprise Management (WBEM), and the Directory-Enabled Networking (DEN) proposed by DMTF for web-based network management (DMTF 2004), which uses the Common Information Model (CIM) (DMTF 1999) for object-oriented description of the management data; (4) The Policy-Based Management (PBM) (Sloman, 1994, Lupu *et al.*, 1999, Daminanou *et al.*, 2001), etc. However, these approaches, some of which are quite complicated, have been developed to deal with the then-existing management tasks and could not always anticipate their evolution and the advent of new challenges. Recent works that address the newer issues in service configuration are: (1) the NETCONF configuration protocol (Enns, 2006); the (2) Meta-CLI Model (Deca *et al.*, 2004, Hallé *et al.*, Oct.2004, Nov.2006); (3) industrial tools (the OPNET's *NetDoctor* tool) (OPNET, 2005); (4) convergent configuration tools (the *Cfengine*) (Burgess, 1995) and algebraic models (Couch *et al.*, 2003), (5) formalisms (Alloy, PAL, TQL, the *Configuration Logic* (Villemaire *et al.*, 2005), (6) ontologies (López de Vergara *et al.*, 2002, 2003), rule integrity logic (Bush *et al.*, 2003, Al-Shaer *et al.*, 2004, Hamed *et al.*, 2006), etc.

To deal with the new challenges, it is necessary to fully understand them and come up with new solutions. In this paper, we focus on several of these main issues, classify them in several categories that we present in the following sections. Section 2 presents

issues due to the hierarchy of service configurations; section 3 analyzes the issues engendered by the distribution of services over multiple equipments of the network; conclusions are formulated in section 4.

2. Hierarchy Issues

The configuration hierarchy concerns the parameters and how they are organized into services, the operations that modify them and the constraints that regulate the existence of the parameters and the execution of operations. The hierarchy constraints may be classified into: (1) constraints among parameters of services that use one another or are components of each other; and (2) constraints among parameters of services that compete with each other. These categories are further refined in the following subsections, after presenting the configuration information and operations context.

Representation of the Configuration Information The set of parameters involved in the deployment or the upgrade of a network service does not have a flat architecture but is structured hierarchically. Figure 1 presents the tree-like structure of a *QoS policy* service example, which contains four configuration components: (1) a *traffic filter*, which defines a filtering policy for the packets; (2) a *traffic grouping*, which classifies and identifies interesting packets using a traffic filter or by checking packet header fields; (3) a *policy*, which decides the QoS functions to be performed); and (4) the *(sub-)interfaces* or *ATM/FR PVCs* on which the policy is applied. Each of these components contains one or more parameters. In this context, finding ade-

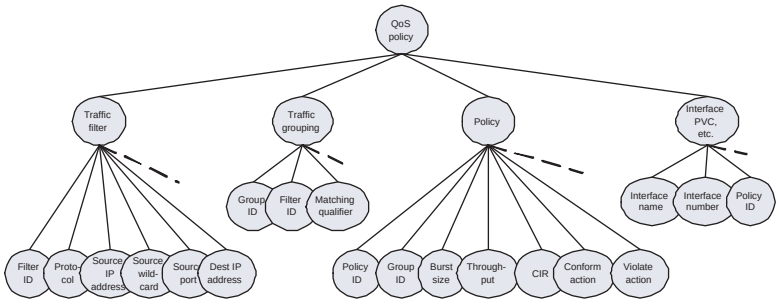


Figure 1. *Example of a network service configuration hierarchy: QoS policy.*

quate representation and modelling of the hierarchy of the service configuration provides multiple abstraction levels of the service configuration, and simplifies the configuration tasks and operations, in the context of increasingly complex, variable and sometimes dynamic service configurations. To solve the complexity introduced by the configuration constraints, these must be embedded in the configuration hierarchy and adequately modelled.

2.1. *Configuration Operations*

To deploy or upgrade some service on a network equipment, the configuration operations may act either directly on the writable configuration parameters or indirectly on the read-only configuration parameters on that equipment. Configuration operations may be followed by, or alternated with, validation or verification operations. Validation may be done at specific points during the procedure by online, or dynamic, validation operations, which may inspect both read-only and writable configuration parameters. Offline, or static, validation may be carried out on writable parameters from plain text configuration files. There are several interesting issues regarding the configuration operations, such as: abstraction, granularity, configuration constraints, order of execution, transaction-orientation, etc.

Abstraction and Granularity Issues Some services may require the configuration of thousands of parameters, distributed over hundreds of equipments. In this context, using sequential per-parameter configuration operation is both slow and error-prone, because of the sheer number of operations and of the distance in space and in time on the establishment of parameters that are coupled by constraints. It is therefore necessary to devise configuration operation primitives whose abstraction and granularity levels would map to hierarchical representation of service configuration. thereby allowing the network engineers to focus on ensuring the quality and correctness of service configuration changes rather than on changing configuration parameters.

Constraint Issues Raising the abstraction level of the configuration operations brings additional issues into the spotlight. One issue is that the operations of various levels of abstraction and granularity should capture the relationships and constraints that bind the parameters. Issues are raised also concerning the proper order of execution of the operations and the means to ensure transaction-oriented atomic updates, especially for network services distributed over multiple equipments.

2.2. *Composition Constraints*

The configuration and the composition of the configuration parameters into network service configuration cannot be done randomly lest the resulting service will be incorrect or inexistent. Among the configuration parameters there are multiple constraints (or dependencies, rules) that must be complied with. These constraints may be applied in different contexts: (1) actively, by configuration operations, to update or reconfigure some parameters of the service configuration; or (2) passively, by validation operations, to determine the correct or incorrect status of the service configuration. For efficient service configuration and to guarantee the correctness of the network service configuration, these constraints must be parameterized, captured and expressed by adequate formalisms, embedded in the representations and the operations of the service configuration, and rigorously checked for completeness and absence of contradictions using inference logic and axiomatic systems. In the following subsections, we present several main configuration constraint types (syntactic, semantic, causal,

consistency, distribution, transaction, interaction, heterogeneity, etc.) and illustrate them with examples by means of *Obligation-Permission-Interdiction* (*OPI*) task trees (Barbuceanu *et al.*, 1998) that have been adapted and extended to represent configuration constrains.

Syntactic Constraints are situated at the basis of the configuration constraint hierarchy, which means that they are local to the configuration parameters or the configuration commands. The syntactic constraints that restrict the values of the parameters may stipulate the formats, the ranges and the admissible values. Some of these constraints are automatically enforced by the system, whereas others are not. Unfortunately often the simple ones are enforced while the complex ones are not. Figure 2 provides an example of a syntactic constraint, found in (OPNET, 2005), which restricts the admissible values of the subnet mask parameter. This constraint is usually not enforced by the system. It further restricts the admissible values (ranging from `0.0.0.0` to `255.255.255.255`), which are already enforced by the system. Syn-

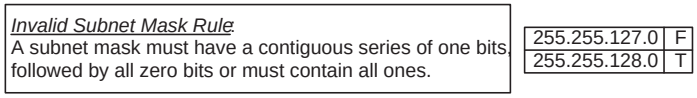


Figure 2. *Syntactic constraint for a configuration parameter: IP invalid subnet mask.*

tactic constraints that restrict the order, the number and the choice of parameters in commands are partially checked by the command parser. However, the complex constraints (e.g. filtering rules that filter both fragmented packets and L4 protocol port information in the packets) are not enforced. These constraints depend on the configuration environment (software and hardware type/version of the network device), due to the fact that the same configuration command may have a different syntax in various CLIs provided by the existing operating systems (Cisco's IOS, Juniper's JUNOS, Lucent's ComOS, etc.). Figure 3 presents the syntax constraints of the IP filter rule command `set filter` in Lucent's ComOS CLI.

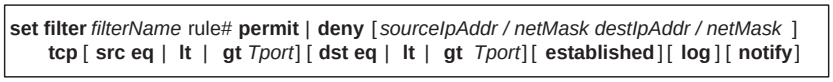


Figure 3. *Syntax of Lucent's* `set filter` *ComOS command.*

Semantic Constraints In a configuration, the existence or the values of some parameters may depend on the existence or the values of other parameters of a service configuration. These semantic constraints express the structural or contents relationships among configuration parameters. Often, simple semantic constraints consist of equality requirements of different parameters' values. Figure 4 shows several semantic constraints in an example of *QoS Policy*. The *Traffic grouping* depends on the

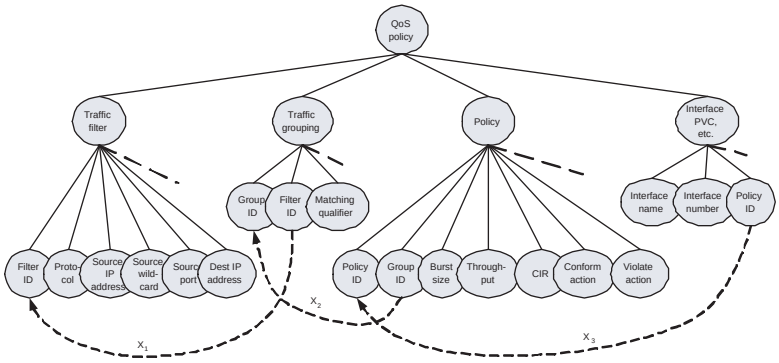


Figure 4. *Example of semantic constraints (X_1, X_2 and X_3) within the configuration of the QoS policy service.*

Traffic filter through constraint X_1, the *Policy* depends on the *Traffic grouping* through constraint X_2, and the *Interface* components depend on the *Policy* component through constraint instances of type X_3. Figure 5 shows a list of semantic constraints used by the OPNET's NetDoctor *IT Guru* tool (OPNET, 2005) to validate VLAN configurations.

VLAN Configuration Rules
1: Peer trunk ports must have the same encapsulation (dot1Q or ISL).
2: Peer trunk ports that have been configured as 802.1Q trunks must have the same native VLAN ID.
3: Peer ports should have the same port duplex mode.
4: Peer ports should have the same port speed setting.
5: Peer switches running VTP should have the same VTP domain name.
6: Peer switches running VTP should have the same VTP password.
7: A VLAN configured on a port must be defined on the device.
8: A device running VTP should have a VTP domain name configured.

Figure 5. *Examples of VLAN configuration rules.*

Causal Constraints[1] occur when the configuration of some parameters is conditioned by the pre-existence of other parameters. Causal constraints are important for determining the order of execution of the configuration operations. Causal constraints may be classified into two categories. In the first case, the order of operations is rigid. By inversing the order of operations, some of the operations may not take effect and the service may not be configured. If the mandatory order of two operations is (A, B) and the operations have been executed in the reverse order (B, A), then operation B has no effect. In the second case, the order of operations is not mandatory but is logical

1. Or temporal, sequential.

and may be derived from the semantic constraints. For instance, normally, one creates a traffic filter first, then invokes it on some subservice or interface. The OPI tree in Figure 6 shows the causal constraints among the QoS policy service components (derived from the semantic constraints X_1, X_2 and X_3 in Figure 4).

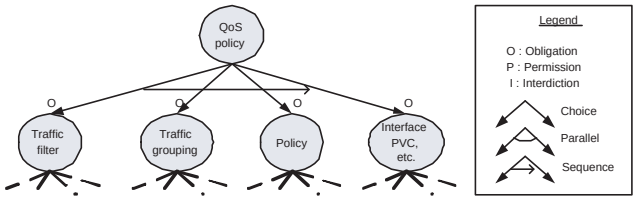


Figure 6. *Example of causal constraints among the components of the QoS policy service configuration.*

Service Consistency and Integrity Constraints Formalizing the constraints with the aid of symbolic notations or languages will not guarantee that the network services are consistent. A configuration may comply with all of the specified constraints and the service may still be inconsistent and lack integrity. To guarantee service consistency and integrity, it is necessary to check whether the set of existing constraints is complete, i.e. there are no missing constraints, and that there are no contradictions among these constraints. For instance, for a traffic filter to be correct, it is required

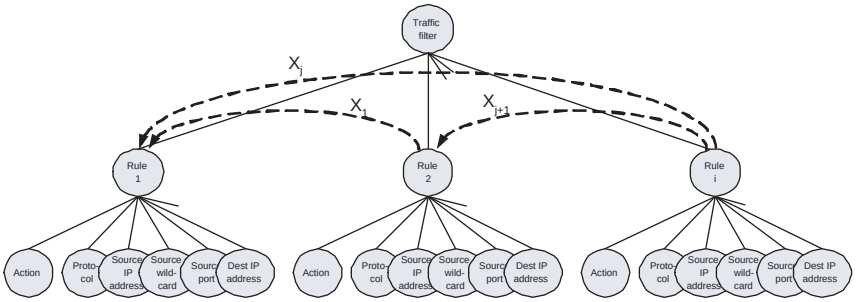


Figure 7. *Example of consistency constraints (X_1, X_j and X_{j+1}) among the configuration parameters of the rules of a traffic filter.*

that each filter rule should not overshadow, nor be redundant with respect to, any of the subsequent rules (Al-Shaer *et al.*, 2004, Hamed *et al.*, 2006). Thus, the parameters of Rule i of the traffic filter shown in Figure 7 should comply with the consistency and the integrity constraints $X_1, ..., X_{j-1}$ that restrain them with respect to the Rules $1, ..., i-1$. Moreover, to guarantee the integrity and consistency, it might be necessary to provide rigorous proofs and demonstrations, using some kind of inference logic, based on axioms, theorems and lemmas.

Component Inter-changeability Constraints The service configuration may have a variable geometry, in which some components are reconfigurable or interchangeable, depending on some inter-changeability constraints. For instance, in the QoS policy configuration (pls. refer to Figure 1), the *Traffic filter* component may be replaced by *packet header bit matching* criteria (which are *Traffic grouping* parameters). This kind of constraints is subtler and more difficult to handle because it may generate a combinatorially large number of parameter combinations and service variants

2.3. *Compatibility and Interaction Issues*

In general, compatibility problems may appear when multiple services are deployed over the same equipment or network. The services may need/use each other or be independent. In the latter case, undesired interactions may occur among the service parameters at customer resources or configuration level. The configuration compatibility issues may arise when the same configuration parameters are required for the configuration of several services. There are several types of configuration compatibility issues: (1) among the multiple services deployed over the same equipment or over the same network, e.g. traffic filter, VLAN, VPN, QoS policy, etc.; (2) among the multiple instances of the same service deployed over the same equipment or over the same network, e.g. different traffic filters assigned to different interfaces; (3) among the multiple features of a service, of multiple instances of a service, etc. The features may be viewed either as concrete instantiations of services (e.g. a traffic filter service must be either inbound or outbound) or as additions to the basic functionality of services (e.g. a traffic filter for non-initial packet fragments), according to whether the parameter that specifies the feature is mandatory or optional, respectively. Figure

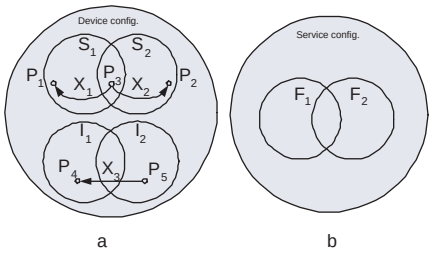


Figure 8. *Types of interactions in service configurations.*

8(a) shows the diagram of the configuration parameters of two services S_1 and S_2 that have some common configuration parameters, such as P_3, and the diagram of two instances I_1 and I_2 of the same service. Figure 8(b) shows the diagram of the configuration parameters of two features F_1 and F_2 that have some common configuration parameters.

Compatibility constraints The constraints that restrict the interacting parameters may be classified in two categories: (1) constraints that bind parameters that are common to the configurations of different services of features; (2) constraints that bind parameters that belong either to one service configuration or the other. The configuration compatibility constraints may be classified in to categories, according to whether the parameters are common to the configuration spaces of the interacting services or not. For instance, in Figure 8(a), constraints X_1 and X_2 belong to the former category, since they restrict parameter P_3 that belongs both to the configurations of service S_1 and service S_2, whereas constraint X_3 belongs to the latter category since neither P_4 nor P_5 is a common parameter. It should be noted that, in the former case, the incompatibility among the competing constraints causes the incompatibility of the services. The service compatibility constraints may be obtained as compositions and extensions of intra-service at a higher, inter-service, level. For instance, by combining the intra-service constraints $X_1(P_1, P_3)$ and $X_2(P_1, P_3)$ is obtained the S_1/S_2 compatibility constraint: $X_4(P_1, P_2, P_3) = X_1(P_1, P_3) \wedge X_2(P_1, P_3)$.

3. Distribution Issues

The distribution of the services may occur at two levels: (1) at local level, some service distribution may be distributed over multiple (sub)interfaces, ports, PVCs, channels, etc. For instance, the same packet filter, QoS policy or VLAN may be activated on several (sub)interfaces of the same equipment; (2) at network level (which is more complex), some service may be distributed over multiple equipments. E.g., several equipments may belong to the same VLAN. In the example shown in Figure 5, the VLAN validation rules 1–6 are distributed over two equipments. In the case of network-level distribution, the order of the operations distributed over equipments may be important and needs to be clearly determined and enforced. This aspect confers a distributed characteristic to the causal constraints.

Transaction Issues Another issue of distribution is ensuring the atomicity, consistency, integrity and durability by providing adequate, distributed, multi-device, primitives for transaction-oriented operations and by correlating the order of execution of these operations on each equipment. Figure 9 shows an ad-hoc combination of an OPI and hierarchy tree representing an example of a scenario using a very basic protocol of transaction-like operations applicable to a service configuration that consists of multiple components and is distributed over multiple equipments. The equipments are locked and then the parameters of the components are configured. After the components are validated, the changes are committed to the equipments (or discarded, should the validation fail) and the equipments unlocked.

Heterogeneity Issues As already mentioned, the network heterogeneity has two causes or dimensions: (1) a spatial dimension, introduced by the distribution of multiple equipments; and (2) a temporal dimension, introduced by the evolution in time of the networks and the services deployed over it. Since, whatever their causes, the heterogeneity issues are similar, we focus here on distribution heterogeneity only. Sev-

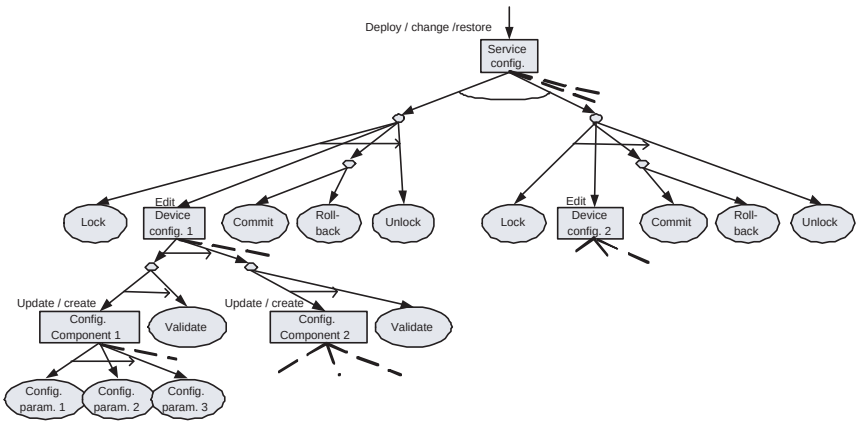


Figure 9. *Transaction constraints of a hierarchical, distributed service configuration.*

eral causes may concur to these changes: (1) variety of equipments; (2) different roles of the equipments, (client/server, edge/core, provider/customer, ingress/egress); (3) network changes: topology, (network outages, congestion), technology, (network upgrades); (4) changes in the customer requirements or in the provider implementation choices; and (5) changes in the network management models. We will focus on the first cause of heterogeneity, which is due to the variety of equipments.

Equipment heterogeneity constraints The services may have different configurations when deployed over equipments characterized by different hardware (vendors, series, types, interfaces) and software (O.S., versions). The heterogeneity constraints may touch all the configuration hierarchy, from parameters to entire services. Thus, the *VLAN trunk encapsulation protocol* parameter takes the value `ISL` [2] for older hardware series and software versions, and the value `DOT1Q` [3] for more recent hardware series and software versions. Similarly, the *Rate limit*, which is an older version of *traffic policy* service, is supported by older equipments and software, whereas the *Class-based policer*, which is a newer version, is supported by newer equipments and software. Figure 10(a) shows the VLAN service configuration as a OPI tree of the trunk encapsulation parameter, whose behaviour is determined by means of environment constraints. When the equipment hardware/software are A, a, the peer interfaces must be configured with the value `trunk encapsulation = ISL` (Figure 10(b)); and (2) when the equipment hardware/software are B, b, the peer interfaces must be configured with the value `trunk encapsulation = DOT1Q` (Figure 10(c)). Note that the configurations in Figures 10(b) and (c) are also constrained by semantic Rule 1 in Figure 5.

2. Corresponding to Cisco's Inter-switch Link trunk encapsulation protocol.
3. Corresponding to IEEE's 802.1Q trunk encapsulation protocol.

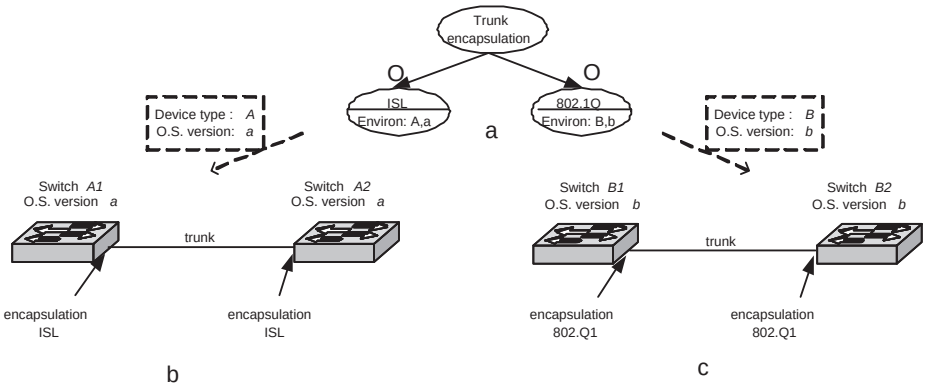


Figure 10. *Example of configuration variation with the equipment environment.*

4. Conclusions

In this paper, we presented several types of constraints in network service configuration. Our analysis shows that the hierarchical constraints, grouped in composition and interaction categories, play a central role in the service configuration issues. Other types of constraints, such as: distribution and heterogeneity, bring additional complexity to these issues. In real life, these constraints build up into composites that are difficult to resolve. For a successful approach to network service configuration, it is therefore necessary not only to find separate solutions to model and resolve each type of constraints, but also to provide some integrative framework for these solutions. The purpose of this paper was to clearly state the issues. Therefore, a discussion and an analysis of possible and existing solutions were left for further work.

5. References

Al-Shaer E. S., Hamed H. H., "Discovery of policy anomalies in distributed firewalls", *Proc. Annual Joint Conference of the IEEE Computer and Communications Societies, (INFOCOM'04)*, March 2004, p. 2605-2616.

Barbuceanu M., Gray T., Mankowski S., "How to make your agents fulfill their obligations", *3rd Conference on the Practical Applications of Intelligent Agents and Multi-Agents (PAAM'98)*, London, UK, 1998.

Burgess M., "Cfengine: a site configuration engine", *USENIX Computing systems*, vol. 8, num. 3, 1995, p. 333-360.

Bush R., Griffin T., "Integrity for Virtual Private Routed Networks", *Proc. 22nd Annual Joint Conference of the IEEE Computer and Communications Societies, (INFOCOM'03)*, San Francisco, CA, U.S.A., 2003.

Couch A. L., Sun Y., “On the algebraic structure of convergence”, *Proc. 14th IFIP/IEEE Internat’l Workshop on Distributed Systems: Operations and Management, (DSOM’03)*.

Damianou N., Dulay N., Lupu E., Sloman M., “The Ponder policy Specification Language”, *Workshop on Policies for Distributed Systems and Networks (Policy’01)*, HP Labs Bristol, 2001, Springer, p. 29-31.

Deca R., Cherkaoui O., Puche D., “Configuration Model for Network Management”, *Proc. IFIP TC6 Conference on Network Control and Engineering for QoS, Security and Mobility (Net-Con’2004)*, Palma de Mallorca, Spain, 2004.

DMTF, “Common Information Model (CIM) Specification. Version 2.2. Specification”, 1999.

DMTF, “Directory Enabled Network (DEN) Initiative”, www.dmtf.org/standards/den, 2004.

Enns R., “NETCONF Configuration Protocol”, http://www.ietf.org/internet-drafts/draft-ietf-netconf-prot-12.txt, 2006, IETF Internet-draft.

Hallé S., Deca R., Cherkaoui O., Villemaire R., “A formal validation model for the netconf protocol”, *Proc. 15th IFIP/IEEE Internat’l Workshop on Distributed Systems: Operations and Management (DSOM’04)*, Davis, CA, USA, November 15-17, 2004, p. 147-158.

Hallé S., Deca R., Cherkaoui O., Villemaire R., “Automated validation of service configuration on network devices”, *7th Internat’l Conference of Multimedia Networks and Services (MMNS’04)*, San Diego, California, USA, October 3-6, 2004.

Hallé S., Deca R., Cherkaoui O., Villemaire R., Puche D., “Sequential dependencies in configuration operations”, *7ème Colloque francophone de Gestion de Réseaux et de Services*, Bordeaux, France, May 9-12, 2006, Hermès, Paris.

Hamed H. H., Al-Shaer E. S., “Taxonomy of Conflicts in Network Security Policies”, *IEEE Communications Magazine*, vol. 44, num. 3, March 2006.

IETF, RFC1357, RFC1351-3, RFC1441, RFC1445, RFC1446, RFC1448, RFC1449.

ISO/IEC, “Information Processing Systems - Open Systems Interconnection - Structure of Management Information. Part 4: Guidelines for the definition of Managed Objects”, 1991.

ITU-T, “Generic Network Information Model”, 1992.

López de Vergara J., Villagrá V. A., Berrocal J., “Semantic Management: advantages of using an ontology-based management information meta-model”, *Proc. HP Openview University Association 9th Plenary Workshop (HP-OUVA’02)*, Böblingen, Germany, 2002.

López de Vergara J., Villagrá V. A., Asensio J. I., Berrocal J., “Ontologies: Giving Semantics to Network Management Models”, *IEEE Network*, vol. 17, num. 3, 2003, p. 15-21.

Lupu E. C., Sloman M., “Conflicts in Policy-Based Distributed Systems management”, vol. 25, 1999, p. 852-869.

OPNET Technologies, Inc., “NetDoctor - Ensuring Network Integrity and Security”, 2005, http://www.opnet.com/products/modules/netdoctor.pdf.

OPNET Technologies, Inc., “NetDoctor User Guide for IT Guru”, 2005, http://www.rh.edu/-webgen/appdocs/opnet/doc/itguru/module_doc/NDr/NDr_23_Rule.pdf.

Sloman M., “Policy Driven Management for Distributed Systems”, *Journal of Network and Systems Management*, vol. 3, 1994, p. 333-360.

Villemaire R., Hallé S., Cherkaoui O., “Configuration logic: a multi-site modal logic”, *Proc. 12th Internat’l Symposium on Temporal Representation and Reasoning, (TIME’05)*, Burlington, Vermont, USA, June 23-25, 2005.

Sequential Dependencies in Configuration Operations

Sylvain Hallé, Rudy Deca, Omar Cherkaoui, Roger Villemaire

Université du Québec à Montréal
C.P. 8888, Succ. Centre-ville
Montréal (Québec) CANADA H3C 3P8
halle@info.uqam.ca

Daniel Puche

Cisco Systems inc.
dpuche@cisco.com

RÉSUMÉ *Le déploiement d'un service réseau est sujet à plusieurs dépendances sémantiques et séquentielles. Cependant, un des principaux problèmes des approches de gestion des configurations est l'absence d'un modèle transactionnel qui permettrait aux informations de configuration de conserver leur intégrité durant le processus de configuration. Dans cet article, nous introduisons la notion de dépendance séquentielle et proposons un modèle mathématique basé sur les techniques du model checking permettant de structurer les opérations de configuration. Ce modèle mène au concept « d'état-borne » (milestone state). Nous suggérons par la suite une manière de bonifier le protocole Netconf avec une composante transactionnelle basée sur ces concepts.*

ABSTRACT. *The deployment of a network service is subject to a number of semantical and sequential dependencies. However, one of the main issues with the existing configuration management approaches is the absence of a transactional model, which should allow the network configuration data to retain their integrity during the configuration process. In this paper, we introduce the notion of sequential dependency, propose a mathematical framework based on model checking that allows the structuring of configuration operations leading to the concept of milestone state, and suggest how the Netconf protocol can be enhanced with a transactional component.*

MOTS-CLÉS: *gestion des configurations, Netconf, dépendances sémantiques, dépendances séquentielles, model checking, LTL*

KEYWORDS: *configuration management, Netconf, semantical dependencies, sequential dependencies, model checking, LTL*

1. Introduction

The deployment and configuration of network services is a complex and error-prone task that is subject to constraints at different levels. For instance, *semantical* dependencies between parameters dispersed among multiple configuration operations appear in even the simplest management tasks (Hallé *et al.*, 2004a). Although these dependencies are not currently captured by management protocols such as Netconf (Enns, 2005), it has been shown how tree logics can help in automating their formalising and checking on a given configuration snapshot (Hallé *et al.*, 2004b).

However, even if semantical dependencies can be automatically verified, an important part of the complexity of deploying a service still remains. In general, the configuration operations must be performed in a specific order that is determined by the connected nature of the network, or even by some requirements of the devices' operating system. Therefore, in addition to semantical dependencies, there are *sequential* dependencies that need to be formalised and checked in a smiliar fashion. The importance of checking these sequential dependencies is heightened by the fact that an increasing number of services, such as Virtual LANs and Virtual Private Networks, involve configuration changes on multiple devices at the same time.

While the semantical dependences in network device configurations have been widely debated in the network management literature (Daminaou *et al.*, 2001; Warmer *et al.*, 1999; Crubézy, 2002; Jackson *et al.*, 2000; Hallé *et al.*, 2004b), the sequential dependences have not yet been extensively covered. Among the works on the subject, (Couch *et al.*, 2003) examine a convergent approach to automated configuration and provide an algebraic model of configuration management. According to this model, the managed processes can be decomposed into regions or intents of non-conflicting, stateless actions. Each of these non-commutative regions can then be processed separately. Using this model, procedural processes, which are composed of non-commutative operations, can be redesigned as declarative processes, which are composed of commutative operations. The authors illustrate their approach with examples from file editing.

The transactional aspect of the device configuration process is taken into account by the Netconf configuration protocol (Enns, 2005), which is a new protocol designed for manipulating network device configurations. However, this protocol provides transactional operations at device level, but does not currently have similar operations for the network level.

The purpose of this paper is twofold. First, we raise the question of sequential constraints in network configuration operations and show by a couple of examples that their presence is as common as semantical ones; using concepts borrowed from the field of model checking, we also demonstrate how these constraints can be accurately formalised in Kripke structures by temporal logic formulas. Second, we

define the notion of *milestone* states in a Kripke structure in terms of these mathematical grounds and claim that these states make good candidates for validation, synchronization and rollback points during the deployment of a service, and illustrate how the Netconf protocol could be enhanced by the addition of a likewise transactional component.

The paper is structured as follows. In section 2, we briefly overview the concepts of configuration tree and semantical dependencies and introduce by the means of concrete examples the concept of sequential dependency. We also describe the mathematical framework of model checking and show how sequential dependencies can be modelled by temporal logic. In section 3, we introduce the concept of milestone and apply it to the Netconf protocol. Finally, section 4 shows some experimental results and section 5 concludes with future paths of work.

2. The Sequential Aspect of Network Management

The deployment of a service over a network basically consists in altering the configuration of one or many equipments to implement the desired functionalities. We can presuppose without loss of generality that all properties of a given configuration are described by attribute-value pairs hierarchically organised in a tree structure (Hallé *et al.*, 2004a; Villemaire *et al.*, 2005) like the one shown in Figure 1.

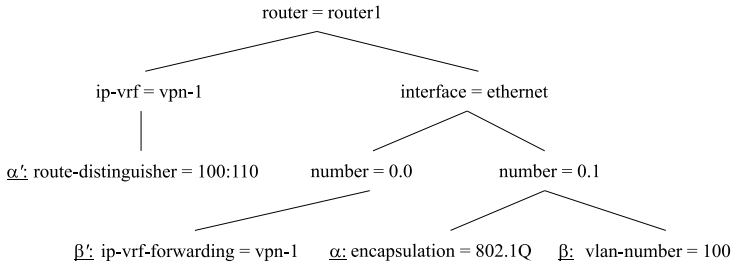


Figure 1. *A sample configuration tree. Nodes labelled α, α', β and β' are not present initially, but are added in the process of deploying the network services given later as examples.*

Possible alterations to the configuration typically include deleting or adding new parameters to the configuration of a device, or changing the value of existing parameters. In most cases, the parameters involved in such modifications are both syntactically and semantically interdependent. For instance, the value of some parameter might be required to depend in a precise way on the value of another parameter; the simplest example of such dependency is the fact that an IP address must match the subnet mask that comes with it. More complex dependencies might

constrain the existence of a parameter to the existence of another. Recent works have shown how such dependencies can be automatically checked by logical tools on a given configuration snapshot (Hallé *et al.*, 2004b).

2.1. *Sequential Dependencies at the Service Level*

However, the situation becomes more complex when one wants to actually *deploy* a service from scratch. In addition to constraints on the values of parameters, the dependencies may also impose that the modifications be performed in a specific order. When done in an uncoordinated way, changing, adding or removing components or data that implement network services can bring the network in an inconsistent or undefined state. This fact becomes acutely true in the case where operations must be distributed on multiple network elements, as they cannot be modified all at once. Moreover, while a single inconsistent device can ultimately be restarted when all else fails, there is no such "restart" option when an entire network configuration becomes inconsistent.

We illustrate the concept of sequential dependencies by the means of two examples taken from the deployment of network services. For each of these examples, a sequential dependency is extracted and formalised.

2.1.1. *Example 1: Virtual LANs*

A Virtual LAN (VLAN) is a group of devices spanning multiple LAN segments that are configured to communicate as if they were connected to the same wire. Each VLAN works as a completely separate entity that can only be joined by a router. Since VLANs are logically (rather than physically) structured, they are extremely flexible. Among the several protocols designed to this purpose, IEEE 802.1Q (IEEE, 1998) has become the standard.

When configuring a router on a VLAN, the subinterface that is connected to a VLAN *trunk* must be set to support the 802.1Q protocol; since each subinterface is attached to a specific VLAN, the number of this VLAN must also be specified when configuring the trunk. Therefore, configuring a VLAN trunk will have for effect of adding nodes α and β in the configuration tree of Figure 1.

However, 802.1Q frames are designed in a way that they must contain the VLAN number; therefore, encapsulation and VLAN number must be configured toghether. From this simple example, one can deduce this first sequential rule:

Sequential Constraint 1 In a router, the VLAN number must be set at the same time the encapsulation protocol is enabled.

In the case of Figure 1, this means that nodes α and β must be added to the tree in the same step.

2.1.2. *Example 2: Virtual Private Networks*

A VPN is a private network constructed within a public network such as a service provider's network (Rosen *et al.*, 1999). A customer might have several sites, which are contiguous parts of the network, dispersed throughout the Internet and would like to link them together by a protected communication. The VPN ensures the connectivity and privacy of the customer's communications between sites.

Such a service consists of multiple configuration operations; in the case of Layer 3 VPNs, it involves setting the routing tables and the VPN forwarding tables, setting the MPLS, BGP and IGP connectivity on multiple equipments having various roles, such as the customer edge (CE), provider edge (PE) and provider core (PC) routers. An average of 10 paramters must be added or changed in each device involved in the deployment of the VPN.

As an example, for a Layer 3 VPN using MPLS, Figure 1 shows two leaf nodes that must be added, each in its own position, to the configuration tree of a PE router. Node α' corresponds to the creation of the Virtual Routing and Forwarding Tables (VRF) necessary for the proper functioning of the VPN; node β' associates this VRF to a specific interface on the router. Semantically, it is clear that one cannot associate a VRF to an interface before the VRF is created in memory. Therefore, trying to add node β' to the configuration before node α' is created is nonsensical and generates an error. From this situation, we can elicit a second sequential rule:

Sequential Constraint 2 To add node `ip-vrf-forwarding` to a configuration tree, the node `route-distinguisher` must already be present.

Special emphasis must be made on the fact that the node `route-distinguisher` has to be present in the tree *before* node `ip-vrf-forwarding` is added, which rules out the possibility that both nodes be added in a single operation.

2.2. *Formalising Sequences of Configuration Operations*

To formalise the sequences of operations, we first need to introduce some basic concepts taken from the theory of model checking (Clarke *et al.*, 2000). Let *S* be a set of *states* representing a unit situation at a given time. In the context of network configuration, states are labelled trees, as described previously.

We call *transition* from a state s_1 to a state s_2 the structural modifications that transform s_1 into s_2. Formally, transitions can be defined as a subset of tuples $T \subseteq S \times S$; there exists a transition from s_1 to s_2 if and only if $(s_1, s_2) \in T$. The tuple (S, T) forms a directed graph *G* that we call a *Kripke structure*. Figure 2 shows an example of a Kripke structure.

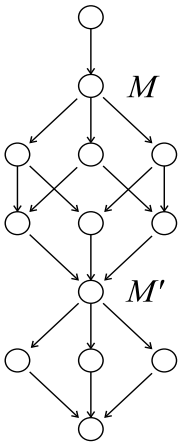


Figure 2. *A Kripke structure with multiple paths from a start state to a target state. Each state represents a labelled tree.*

In the case of the labelled trees we use for modelling device configurations, structural modifications are limited to addition of a labelled node to a leaf, deletion of a leaf node and change in a node's value. These modifications intuitively refer to addition, deletion or modification of a parameter in the configuration of a device.

A *path* is a finite sequence of states $\langle s_1,\ldots,s_n\rangle$ such that, for any s_i, s_{i+1}, there exists a $t \in T$ such that $t=(s_i, s_{i+1})$.

The deployment of a service is a path in such a structure that starts from a given configuration, s_s, and ends at a target configuration s_t. For example, in the case of Figure 1, a possible start state could be the tree without any of α, α', β, β', and a possible target state could be the same tree with all these nodes. A valid deployment sequence could be a sequence of addition of the nodes that respects, among other things, Sequential Rules 1 and 2.

2.3. *Formalising Sequential Dependencies*

The state space generated by spanning all possible transitions between a start and target state is fairly large. For the 4 nodes of Figure 1, there are 24 possible unconstrained paths, and in general, for *n* possible operations, there are *n*! possible paths. We must now restrict our study to acceptable paths —that is, paths that respect the elicited sequential constraints. For this purpose, we use the Linear Temporal Logic (LTL) commonly used in model checking (Clarke *et al.*, 2000).

LTL is a logic aimed at describing sequential properties along paths in a given Kripke structure. Its syntax is based on classical propositional logic, to which modal path operators have been added.

The first such modal operator is **G**, which means "globally". Formally, the formula **G** φ is true on a given path π when, for all states along this path, the formula φ is true. The second modal operator commonly used is **X** ("next"). The formula **X** φ is true on a given path π of the Kripke structure when the next state along π satisfies φ . Finally, a formula of the form **F** φ ("eventually") is true on a path when at least one state of the path satisfies φ . Other modal operators exist, but go beyond the scope of this paper.

A LTL formula is a well-formed combination of the classical $\wedge$ (disjunction), $\vee$ (conjunction), $\neg$ (negation), $\rightarrow$ (implication) and $\leftrightarrow$ (equivalence) operators with modal operators. The *atoms* of LTL are the base-level Boolean expressions over which the formulas are built. In the present case, since states are labelled trees, we take the atoms to be formulas themselves, based on a tree logic such as CL (Villemaire *et al.*, 2005).

Equipped with it, it is now possible to formalize sequential constraints into logical formulas. Without delving into further details, suppose that φ_n is a CL formula that is true if and only if node *n* is present in a given configuration tree. Then, the Sequential Constraint 1 presented in the previous section can be translated into the following formula:

Sequential Formula 1

$$\mathbf{G}\ \ \varphi_\alpha \leftrightarrow \varphi_\beta$$

This formula means that in all states of all paths where either α or β exists, the other node must also exist. All temporal rules described earlier can therefore be translated into LTL formulas of that kind. For example, Sequential Rule 2 becomes:

Sequential Formula 2

$$\mathbf{G}\ (\ \neg\varphi_{\alpha'} \rightarrow \mathbf{X}\ (\ \varphi_{\alpha'} \rightarrow\ \neg\varphi_{\beta'}\ ))$$

telling that the presence of node β' implies that node α' is present, and that is must also have been present at least in the previous step in the deployment.

3. Transactional Aspects of the Netconf Protocol

We now show how transactional aspects presented in the previous section can be applied to the Netconf protocol by enhancing it with a transactional component.

3.1. *Overview of the Netconf Protocol*

To send configuration commands to a router, Netconf provides a set of "remote procedure calls" (RPC) and RPC-replies. In a simplified way, an RPC is a block of

XML data whose opening tag contains an identifier that either asks the router to return a portion of its configuration file, or tells it to replace a part of its configuration with a snippet provided by the user and carried in the body of the RPC. Netconf offers other built-in operations, such as commands allowing to lock a part of the router's configuration so that only the current user can modify it, and subsequently unlock it. The RPC-reply is the XML block that is returned to the user.

The Netconf protocol defines a simple mechanism for device management. However, its transactional model, which includes a validation capability, is device-centered, and does not provide a mechanism to ensure the consistency of the sequence of operations with respect to the rules elicited in section 2.

In order to bestow transactional semantics on the update operations of multiple configurations, it is important to determine the optimal points of validation, commitment and roll-back during the update process of the network device configurations.

3.2. *Components and Milestones*

We propose to determine these points by analysing special properties of the Kripke structure induced by the sequential dependencies. To this purpose, we introduce the notion of *milestone state*. A milestone state is a state *m* by which all valid paths must eventually pass. Formally, let ψ be some LTL temporal rule, and τ_m be a CL formula that is true only on state *m*. Then, in a Kripke structure *G*, *m* is such that for every path beginning at the start state and that satisfies ψ, the formula $\mathbf{F}\ \tau_m$ is true.

Milestones can be thought of as unavoidable steps in the path from start to solution, since all acceptable paths must eventually pass by those points, in the order they appear. In the case of Figure 2, we see two milestones, labelled *M* and *M'*.

We argue that milestones are good candidates to divide the modelled process into natural macro-steps of which they are the boundaries; complementary to milestone states are *components*, i.e. sets of states and transitions comprised between two milestone states. The word “natural” is used here, since these milestones emerge from the set of temporal constraints imposed on the lattice. Different temporal constraints generally lead to different milestones.

3.3 *Towards a Transactional Model*

The main advantage of the analysis of the lattice that arises from temporal constraints is that it induces a way of synthesising a protocol for the

implementation of a service. By placing validation checkpoints at milestones, we ensure such checkpoints are placed in semantically sound locations throughout the deployment process. Since these checkpoints reflect the structure imposed by the temporal constraints, they also make good points to roll back in case a failure occurs.

These points are important since they represent optimal places of validation in the flow of operations. Thus, a validation in one of these points can check all or most of the dependences that apply on the multiple flow streams that converge towards the validation point. Moreover, these convergence points are unavoidable during the configuration and, since they concentrate the flow paths, a validation performed at such points provides a maximum extent of coverage for those flows.

Intuitively, we suggest that a validation point be used to validate the operations that are situated along the flow path connecting it to a previous upstream milestone, in which a validation has been already done. If the validation has been successful, the update information generated by the operations is committed. Otherwise, if the validation or the commitment fails, the update information generated by the operations is discarded and the configurations are rolled back to the latest points of successful validation.

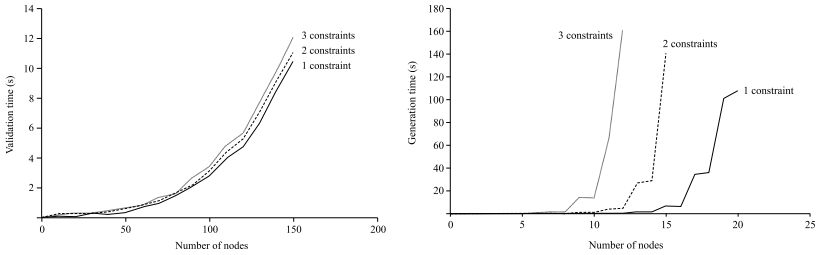
Netconf provides two phases of a successful configuration transaction during a service configuration procedure: preparation and commitment. During preparation, the configurations are retrieved from the network devices. When all the configurations have been retrieved, the edition starts at service level. The validation at this stage ensures that the network configuration is consistent before the proposed modifications required by the service. To ensure the integrity of the configuration edition, the device configurations are locked, edited and subsequently unlocked. When the service edition has been successfully accomplished, the commitment starts. The validation at this stage ensures that the network configuration remains consistent after the respective modifications of the network configurations.

Since the network service update affects multiple device configurations, a two-phase commit is required. The first phase stores the update information on temporary storage and validates it before entering the second phase. If the validation is successful, the update information is transferred onto the real configurations, otherwise this information is discarded. In case of erroneous transfers during the second phase, the configurations are rolled back and the second phase can be resumed.

This semantics can be used with the Netconf configuration protocol to ensure the transactional properties of the service updates on multiple devices. As already mentioned, the Netconf protocol defines transactional operations for device level but does not provide similar operations for network level, i.e. for the multiple device configurations supporting a network service.

Obviously, the higher-level validations may involve multiple devices. For instance, the routing table and the protocol information in a device depend on the network addresses and the protocol information from other devices. Similarly, parameters such as protocol neighbors' IP addresses, autonomous system numbers, protocols' process number and IP addresses must accurately correspond on more than one device, in order for the network service that is deployed over that network to be consistent.

In this case, defining an operation that can validate multiple parameters situated on several devices might be highly recommendable. This multi-device validation operation would replace multiple single-device validation operations and would allow performing complex validation queries directly within the given configuration protocol.



a) b)

Figure 3. a) *Validation time of a deployment sequence in terms of number of nodes to alter and constraints per node.* **b)** *Generation time of a valid deployment sequence in terms of number of nodes to alter and constraints per node.*

4. Experimental Results

Since the structures generated by sevice deployments are Kripke structures and that the sequential formulas can be formalised in LTL, it is possible to submit the problem directly to a model checker like NuSMV (Cimatti *et al.*, 2002). Using this tool, we generated sample deployment sequences and checked that these deployments respected a set of constraints similar to Sequential Rules 1 and 2. For each test, we varied the number of nodes in the sequence and the number of sequential constraints imposed on each nodes. The validation times for these experiments are summarised in Figure 3a. All times given in this section have been calculated on an AMD Athlon XP-M 2200+ running NuSMV 2.1.2 under Cygwin.

One can see that validation times for large sequences of operations (up to 150 nodes) remain under the reasonable bound of 10 seconds, and that augmenting the number of constraints is not the principal factor that makes the computation longer.

Additionally, it is possible to benefit from the counter-example generation mechanism of NuSMV to find a deployment sequence that does not violate any constraint. As a matter of fact, when a LTL property of the form **G** *p* is false, NuSMV provides the user with an execution trace on the Kripke structure for which *p* is false. If *p* is the LTL property one wants to verify on a structure, it suffices to submit the formula **G** ¬*p* for verification. If there exists a trace for which *p* is true, then such a trace is a counter-example for the formula **G** ¬*p*, and therefore NuSMV will display it to the user, giving by the same occasion a valid deployment sequence.

We have conducted experiments with NuSMV on sample deployment sequences with constraints of the same form as Sequential Formulas 1 and 2. We varied the size of the configurations and the number of sequential constraints per node imposed on the structure, and computed the time NuSMV took to provide a correct deployment sequence. The results of these experiments are presented in Figure 3b. Each curve corresponds to the generation time of a valid deployment sequence involving some number of nodes, with 1, 2 or 3 sequential constraints imposed on each *node* —that is, the total number of constraints actually increases with the number of nodes.

As expected, generating a valid sequence is much harder than validating an existing one. Moreover, the number of sequential constraints on each node does matter in this case, and can change a rather simple situation into an untractable one. One can see that, for sequences that involve the addition or modification of about 10 nodes, which is comparable to deployment of a simple VPN on a router, up to three sequential constraints per node can be imposed without the generation time becoming prohibitive.

These findings suggest that model checking is indeed an interesting tool for on-the-fly validation of deployment sequences, and for offline, *a priori* synthesis of valid sequences for network services with a complexity comparable to a Virtual Private Network.

5. Conclusion

In this paper, we have shown how Linear Temporal Logic applied to Kripke structures can accurately formalise sequential constraints in the deployment of network services. Using these model checking concepts, we defined the notion of *milestone* states in a Kripke structure and gave arguments for using these points as validation, synchronization and rollback points during the deployment of a service,and illustrated how the Netconf protocol could be enhanced by the addition of a transactional component based on milestones.

Empirical results on sample network configurations demonstrate the feasibility of validating deployment sequences using model checking tools, and show that

finding a deployment sequence that validates a set of constraints is a computationally hard problem.

The authors plan future work on these concepts in order to further use milestones in a hiearchical decomposition of a service deployment. In such a setting, each component could contain sub-milestones that would further divide a process into sub-steps based on the same principle. Moreover, the current methodology could be extended by considering all possible orderings of operations in a component and eventually reduce the study to one specific ordering, in the same way *partial order reduction* reduces the state space in model checking (Clarke *et al.*, 2000).

References

Cimatti, A., Clarke, E. M., Giunchiglia, E., Giunchiglia, F., Pistore, M., Roveri, M., Sebastiani, R., Tacchella, A: NuSMV 2: An OpenSource Tool for Symbolic Model Checking. Proc. International Conference on Computer-Aided Verification (CAV 2002), 359-364. (2002)

Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking, MIT Press (2000)

Couch, A., Sun, Y.: On the Algebraic Structure of Convergence, Proc. DSOM 2003, Springer, 28-40 (2003)

Crubézy, M.: The Protégé Axiom Language and Toolset ("PAL"). Protégé Project, Stanford University (2002) http://protege.stanford.edu/

Daminaou, N., Dulay, N., Lupu, E., Sloman, M.: The Ponder policy Specification Language, Proc. Policy'2001, Springer, 29-31 (2001)

Enns, R.: Netconf Configuration Protocol. Internet draft, April 2005. http://www.ietf.org/internet-drafts/draft-ietf-netconf-prot-06.txt

Hallé, S., Deca, R., Cherkaoui, O., Villemaire, R.: Automated Verification of Service Configuration on Network Devices. Proc. MMNS 2004, Springer, 176-188 (2004).

Hallé, S., Deca, R., Cherkaoui, O., Villemaire, R., Puche, D.: A Formal Validation Model for the Netconf Protocol. Proc. DSOM 2004, Springer, 147-158 (2004)

IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks. IEEE Standard 802.1Q-1998 (1998)

Jackson, D., Schechter, I., Shlyakhter, I.: Alcoa: the Alloy Constraint Analyzer, Proc. ICSE (2000)

Rosen, E., Rechter, Y.: BGP/MPLS VPNs. RFC 2547 (1999)

Villemaire, R., Hallé, S., Cherkaoui, O.: Configuration Logic: A Multi-site Modal Logic. Proc. TIME 2005, IEEE Computer Society, 131-137 (2005)

Warmer, J., Kleppe, A.: OCL: The constraint language of the UML. Journal of Object-Oriented Programming, May 1999.

SESSION 3

LES RESEAUX MOBILES

Formation de grappes minimisant l'énergie dans les réseaux de capteurs

Omar Moussaoui, Adlen Ksentini, Mohamed Naïmi

Laboratoire LICP, Université de Cergy-Pontoise
2 avenue Adolphe Chauvin, BP 222, 95302 Cergy-Pontoise cedex

{omar.moussaoui, adlen.ksentini, mohamed.naimi}@dept-info.u-cergy.fr

Mourad Gueroui

Laboratoire PRISM, Université de Versailles Saint Quentin en Yvelines
45 Avenue des états unis, 78035 Versailles cedex

mogue@ prism.uvsq.fr

RÉSUMÉ. *La gestion de la consommation d'énergie est une opération essentielle pour prolonger la durée de vie du réseau de capteurs sans fil (RCSF). Dans ce contexte, les algorithmes de formation de grappes (clustering) ont démontré une meilleure efficacité pour réduire la consommation d'énergie des capteurs. Dans cet article nous proposons un nouvel algorithme de formation de grappes distribué et non périodique. Cet algorithme prend en considération la taille des grappes, la puissance de transmission ainsi que la réserve d'énergie des nœuds (capteurs) pour diviser le RCSF en grappes locales, équilibrées et non recouvertes. Dans le but de minimiser le surcoût du calcul et de la communication, l'algorithme proposé exécute la procédure de formation de grappes seulement à l'activation du système, tandis que la réélection des CHs (Cluster Heads) est aussi longtemps que possible retardée. Cet algorithme réalise également une distribution assez uniforme de CHs à travers le RCSF.*

ABSTRACT. *Energy efficiency operations are essential in extending Wireless Sensor Networks lifetime. Among the energy-saving-based solutions, clustering sensor nodes is an interesting alternative that features a reduction in energy consumption. This paper introduces a novel clustering algorithm that uses a distributed approach to set up non-overlapping clusters, while performing cluster heads rotation as well as carrying out other energy intensives tasks. Besides the transmission power and the energy level of each node, the proposed clustering algorithm considers nodes number that a cluster head can handle aiming at ideally partitioning the network. Further, intending to reduce the computation and the communications cost as well as the messages exchanged, the proposed algorithm invokes the cluster formation only at the system activation, and delays as long as possible the cluster head turn over.*

MOTS-CLÉS : *réseau de capteurs, formation de grappes, économie d'énergie, durée de vie du réseau.*

KEYWORDS: *sensor networks, clustering, energy efficiency, network lifetime.*

1. Introduction

Le réseau de capteurs sans fil (RCSF) est une technologie émergente qui vise à offrir des capacités innovantes de surveillance, d'instrumentation et de contrôle du monde physique (I. Akyildiz et al. 2002). Le RCSF se compose, en général, d'un grand nombre de dispositifs (capteurs) à communication sans fil, qui sont peu coûteux et qui possèdent des capacités d'acquisition, de traitement local de données et une certaine réserve d'énergie. Les capteurs utilisent typiquement des batteries et sont équipés d'un microcontrôleur, d'une mémoire et d'une variété de dispositifs de mesure, comme les dispositifs acoustiques, météorologiques, sismiques et infrarouges. Ces nœuds sont, en général, conçus en petite dimension et cette limitation de taille impose des restrictions aux ressources des nœuds, comme la source d'énergie et les capacités de traitement de données et de communication. La spécificité des applications du RCSF fait que la recharge ou le remplacement de la batterie d'un capteur est une tâche difficile voire même impossible. Cela nous amène à en déduire que la durée de vie d'un capteur est essentiellement dépendante de l'autonomie de sa batterie. Ainsi, la méthode de gestion de consommation d'énergie constitue une contrainte majeure dans ce type de réseau.

Le RCSF peut être utilisé pour diverses applications (militaires, industrielles, biomédicales, environnementales, …). Des nœuds capteurs peuvent être, par exemple, déployés sur des régions lointaines (océans, volcans, fleuves, forêts, etc.) et s'organisent en un seul réseau ad hoc sans fils qui collecte des données pertinentes, exécute le traitement local et dissémine des informations à une ou plusieurs stations de bases. Les domaines d'applications du RCSF sont différents mais les taches exécutées par les capteurs sont les mêmes. Un capteur assure l'acquisition, le traitement de données et la communication. C'est cette dernière tâche qui est la plus consommatrice d'énergie. C'est ainsi que de nombreux travaux portent aujourd'hui sur la gestion de l'énergie des capteurs en considérant en premier lieu la communication entre ces derniers.

Plusieurs techniques en vue de l'optimisation de l'énergie peuvent intervenir dans la mise en place d'un réseau de capteurs. La formation de grappes est une technique prometteuse en raison des avantages qu'elle apporte au RCSF. En effet, l'agrégation des capteurs en grappes permet de réduire la complexité des algorithmes de routage, d'optimiser la ressource médium en la faisant gérer localement par un chef de grappe, de faciliter l'agrégation des données, de simplifier la gestion du réseau, en particulier l'affectation d'adresses, d'optimiser les dépenses d'énergie, et enfin de rendre le réseau plus évolutif (scalable). L'utilisation des grappes permet aussi de stabiliser la topologie si les tailles des grappes sont plus grandes par rapport aux vitesses des nœuds.

La plupart des algorithmes de formation de grappes proposés dans la littérature se basent sur l'élection d'un ensemble de chefs (CH : Cluster Head) parmi les nœuds capteurs du réseau. Par la suite, chaque CH forme sa grappe avec ses voisins qui n'ont pas été élus. De ce fait, les CHs sont responsables de leurs grappes. En effet, d'une part, ils assurent la coordination entre les nœuds de leurs propres grappes (communication intra-grappe) ; puis d'autre part, ils acheminent les données vers les autres CHs (communication inter-grappe). Avec la technique de formation de grappes, les nœuds d'une même grappe transmettent les informations détectées à leur CH et ce dernier se

charge du reste. Il effectue l'agrégation des données récoltées par les capteurs de sa grappe afin d'enlever la redondance et renvoie les données agrégées à la station de base.

Actuellement, le choix des CHs et la formation de grappes, d'une façon optimale, est un problème NP-complet (0S. Basagni et al., 1997). Par conséquent, les solutions existantes à ce problème se basent sur des approches heuristiques et aucune d'entre elles n'assure la stabilité de la topologie du réseau. Evidemment, un bon algorithme de formation de grappes doit autant que possible préserver sa structure lorsque la topologie du réseau change. En effet, la réélection des CHs et l'échange des messages dus à la reconfiguration périodique des grappes impliquent un surcoût de consommation d'énergie très élevé.

Dans cet article, nous proposons un nouvel algorithme de formation de grappes minimisant la consommation d'énergie dans le réseau de capteurs sans fil. Cet algorithme partitionne le RCSF en grappes non recouvertes en se basant sur les propriétés suivantes :

(i) Equilibrer la taille des grappes (nombre de capteurs dans chaque grappe) : aucune des grappes ne doit être sous-chargée ou surchargée. Une grappe de petite taille mène à une mauvaise allocation des ressources alors qu'une grappe de grande taille augmente la latence des nœuds pour accéder aux ressources partagées (comme dans TDMA).

(ii) Minimiser la distance entre les nœuds appartenant à la même grappe : cela permet d'améliorer la qualité de la communication intra-grappe en réduisant aussi bien les interférences que la consommation d'énergie.

(iii) Optimiser la consommation d'énergie de chaque nœud afin d'augmenter la durée de vie du réseau entier.

Une fois les grappes formées, le nœud qui a la plus grande réserve d'énergie parmi l'ensemble des nœuds de chaque grappe devient CH. Puisque les CHs consomment beaucoup plus d'énergie que des nœuds ordinaires et que leur réserve d'énergie est limitée, la réélection des CHs est donc inévitable dans le RCSF. A la différence des autres algorithmes de formation de grappes existants, où la réélection des CHs ainsi que la formation de grappes sont périodiques, notre algorithme maintient les mêmes grappes formées à l'activation du système, et change seulement les CHs d'une manière adaptative. En effet, une fois que le niveau d'énergie d'un CH devient plus petit que la moyenne du niveau d'énergie de tous les nœuds de sa grappe, ce CH cède son rôle au nœud qui a la plus grande réserve d'énergie dans le groupe.

La suite de cet article est structurée comme suit : La section 2 décrit brièvement les algorithmes de formation de grappes existants dans la littérature. Dans la section 3, nous introduisons notre algorithme de formation de grappes qui à comme objectif principal la minimisation de la consommation d'énergie des capteurs. La section 4 présente les performances de cet algorithme par des simulations. Enfin, nous concluons dans la section 5.

2. Etat de l'art et motivation

La façon la plus simple de construire des grappes, outre l'utilisation de l'information géographique, est de choisir comme CH le nœud qui a le plus petit identificateur. Dans 0

(T. Kwon et al., 1999) les auteurs se basent sur cette idée afin d'introduire un algorithme distribué qui permet de construire des grappes à deux sauts. Le but de ces auteurs est de permettre la réutilisation spatiale de la bande passante par la formation de grappes et de contrôler la bande passante dans chaque grappe. Constituer des grappes à deux sauts permet en effet de réutiliser les algorithmes de contrôle de puissance définis pour les réseaux cellulaires (C. Lin et al., 1997).

En général, les algorithmes de formation de grappes se décomposent en deux phases : *(i)* une phase de constitution des grappes ; *(ii)* une phase de maintenance, particulièrement importante lorsque les nœuds sont mobiles. Souvent, la première phase suppose que les nœuds sont quasiment immobiles. Pour pouvoir outre passer cette supposition, l'auteur de (S. Basagni, 1999) a proposé l'algorithme DMAC (Distributed Mobility Adaptive Clustering) qui associe un poids à chaque nœud, tel que d'une part, les CHs obtiennent les poids les plus élevés ; et d'autre part, ces CHs ne peuvent jamais être voisins immédiats. Le poids est variable, et dépend de la vitesse ou de la puissance de transmission.

Les auteurs de LEACH 0(W. Heinzelman et al., 2002) proposent une architecture auto configurable basée sur les grappes afin de minimiser l'énergie des nœuds. Les CHs transmettent directement les données à une station de base, et dépensent donc plus d'énergie que les autres nœuds. Pour répartir la consommation d'énergie, ils suggèrent un algorithme où tout nœud devient régulièrement CH avec une probabilité qui augmente avec le temps passé depuis la dernière fois où le nœud a été élu. De plus cette probabilité est choisie de telle sorte que le nombre moyen de grappes soit un paramètre de l'algorithme. Etant donné que cet algorithme ne garantit pas le nombre de CHs prévu à l'initialisation de l'algorithme ni la distribution équitable des CHs, une version centralisée de l'algorithme (LEACH-C) est donc proposée. Cette dernière permet de déterminer, à partir de la position exacte des nœuds, la configuration optimale pour minimiser l'énergie dépensée. Cependant, la version centralisée de LEACH n'est pas adaptée aux réseaux de grande dimension.

Pour résoudre le problème posé par LEACH sur l'absence de garantie sur la bonne formation des grappes, les auteurs de (O. Younis et al., 2004) proposent l'algorithme HEED qui permet de sélectionner un CH en fonction de son énergie et d'une fonction de coût prédéfinie, basée sur le nombre de ses voisins ou de la moyenne de puissance minimale nécessaire à ses voisins pour l'atteindre. Par conséquent, soit cet algorithme construit un ensemble de grappes très denses, soit, au contraire, forme des grappes se répartissant la charge.

La surconsommation d'énergie, due à la procédure de réélection des CHs ainsi que la reconfiguration des grappes, rend les algorithmes de formation de grappes existant inadaptables au RCSF dans lequel la consommation d'énergie est une contrainte majeure. De plus, aucun de ces algorithmes ne peut assurer une distribution équitable des CHs dans l'espace. En effet, ces algorithmes ne prennent pas en considération la position géographique des nœuds pour élire les CHs. Dans les algorithmes de formation de grappes existant, l'élection des CHs est basée sur l'un des attributs suivants : *(i)* l'identification des nœuds, *(ii)* le poids des nœuds calculé en fonction de certains paramètres comme l'énergie, *(iii)* l'aspect aléatoire. D'ailleurs, dans les algorithmes existants, les nœuds

ordinaires joignent le CH le plus proche sans aucun contrôle sur la taille des grappes. Par conséquent, cela peut engendrer des grappes sous chargées ou des grappes surchargées.

Dans le but de minimiser la consommation d'énergie dans le RCSF, nous proposons un nouvel algorithme de formation de grappes distribué et non périodique. Cet algorithme permet aux nœuds de s'organiser en grappes locales équitables et non recouvertes en se basant sur la proximité des nœuds, la réserve d'énergie des nœuds ainsi que le nombre possible de nœuds qu'une grappe peut contenir. Le groupement des nœuds de proximité dans une même grappe permet aussi bien d'améliorer l'agrégation et la compression des données que la communication intra-grappe.

3. Algorithme de formation de grappes proposé

Dans cette section, nous introduisons les contraintes de construction de grappes ainsi que les objectifs de l'algorithme que nous avons proposé avant de le voir en détail.

3.1 Modèle du réseau

Le réseau de capteurs se compose d'un grand nombre de dispositifs (capteurs) à communication sans fil, lesquels sont dispersés dans un champ rectangulaire. Dans ce réseau, les taches d'acquisition des données sont déclenchées périodiquement ou selon des requêtes envoyées par la station de base. De ce fait, nous considérons un cadre générique pour différentes applications du réseau de capteurs. Nous supposons les propriétés suivantes concernant le modèle du réseau de capteurs:

- les nœuds (capteurs) sont quasiment stationnaires. Cette supposition est typiquement adaptée au RCSF.
- Les nœuds ont les mêmes capacités de capture, de traitement et de communication.
- Les nœuds utilisent une antenne omnidirectionnelle ayant un nombre fixe de niveaux de transmission, c'est-à-dire que les nœuds peuvent régler leur puissance de transmission radio.
- La portée de capture (sensing) d'un nœud est plus petite que la portée de communication.
- Les nœuds se rendent compte de leur voisinage direct (1-saut) et sont capables de mesurer des distances par rapport à leurs voisins immédiats. Ces deux suppositions sont assez raisonnables du fait que la première est une supposition standard pour beaucoup d'algorithmes de découvertes de voisinages, tandis que la deuxième est une caractéristique commune de la plupart des applications du RCSF. En effet, des mesures très précises de calcul de distance entre les nœuds sont possibles dans un RCSF grâce à l'utilisation des systèmes tels que : *(i)* ultrasons (A. Harter et al., 1997), de MIT Crickets (N. Priyantha et al., 2000) ou de Médusa MK-2 (A. Savvides et al., 2003); *(ii)* Ubisense dans les systèmes d'ultra large bande (Ubisense website).

- Des protocoles appropriés de control de transmission pour l'accès au canal sont disponibles pour le RCSF. Ces protocoles doivent fournir une écoute du canal efficace, un mécanisme de back-off aléatoire, un control de contention fiable et un mécanisme qui permet d'éviter le problème des nœuds cachés.

3.2 Contraintes de formation de grappes

En général, le RCSF est constitué par des nœuds et des liens qui peuvent être représentés par un graphe non orienté $G=(V,E)$, où V représente l'ensemble des nœuds et E représente l'ensemble des liens. La division du RCSF en grappes revient donc au problème de partitionnement du graphe avec quelques contraintes supplémentaires. Cependant, la division du graphe en sous graphes connexes de façon optimale est un problème NP-complet. De ce fait, nous nous baserons sur une solution heuristique afin de diviser le RCSF en grappes locales non recouvertes et équilibrées, où chaque groupe contient un certain nombre de nœuds qui seront représentés par un CH. Dans ce contexte, les besoins suivants doivent être répondus :

- La procédure de formation de grappes est complètement distribuée.
- La procédure de formation de grappes se termine dans un nombre fixe d'itérations (indépendamment du diamètre de réseau).
- Lorsque la procédure de formation de grappes est parachevée, chaque nœud appartient à une seule et unique grappe.
- Le résultat de la division d'un réseau de capteur sans fil, représenté par un graphe connexe, en grappes est un ensemble de sous graphes connexes, lesquels sont connectés entre eux.
- La procédure de formation de grappes est efficace aussi bien en terme de complexité de traitement qu'en échange de messages.
- Les CHs sont uniformément distribués à travers le RCSF.

3.3 Objectifs désirés de formation de grappes

Dans le but d'assurer la fiabilité de l'algorithme de formation de grappes, le mécanisme de division du RCSF en grappes doit satisfaire les propriétés suivantes :

- Afin de limiter le nombre d'états à maintenir par le CH et de réaliser une communication intra-grappe efficace, chaque grappe doit contenir un nombre de nœuds préalablement défini. D'une part, les grappes de petites tailles mènent à une mauvaise utilisation des ressources. D'autre part, les grappes de grandes tailles augmentent la latence, qui représente le délai mis par un nœud pour accéder aux ressources partagées (trame TDMA, canal, …) de sa grappe. Par conséquent, le meilleur compromis consisterait à fixer la taille de toutes les grappes par le biais d'un seuil χ à ne pas dépasser. Ainsi ce seuil permettrait de maintenir une charge optimale.
- Sachant que l'énergie consommée due à la communication entre deux nœuds est proportionnelle à la distance entre ces derniers, il est donc nécessaire de limiter

la portée de transmission de chaque nœud à une certaine distance. Autrement dit, la communication à courte distance permet une communication intra-grappe dans de meilleures conditions (atténuation de signal très faible, interférence basse).

- Les nœuds de même voisinage doivent être regroupés dans une même grappe car, en général, les nœuds voisins détectent les mêmes informations. Cela permet de supprimer localement les données redondantes par le CH et de remonter seulement les données pertinentes à la station de base.
- Chaque nœud doit appartenir à une seule et unique grappe afin de minimiser la surconsommation d'énergie. En effet, lorsqu'un nœud appartient à plus d'une grappe, l'énergie est gaspillée par le maintien des états ainsi que l'exécution des taches pour chacune des grappes au lieu d'une seule.
- Afin de réduire les mises à jour du système et de minimiser la surconsommation d'énergie, la reconfiguration des grappes doit être aussi longtemps que possible retardée.

3.4 Description détaillée de l'algorithme proposé

En prenant compte des discussions précédentes, nous proposons un nouvel algorithme de formation de grappes qui combine les paramètres du système ci-dessus (la taille des grappes, la puissance de transmission et l'énergie des nœuds) avec certains facteurs de poids choisis selon les besoins du système. Par exemple, si nous considérons que la puissance de transmission est un paramètre crucial dans le RCSF, alors le poids correspondant à ce paramètre devrait être élevé. De plus, suivant les caractéristiques des applications, un ou plusieurs de ces paramètres peuvent être employés dans le processus de groupement des nœuds dans des grappes.

Dans l'algorithme proposé, seuls les nœuds ayant suffisamment d'énergie peuvent être choisis comme CHs, alors que ceux qui n'ont pas assez d'énergie prolongent leur durée de vie par la réalisation de taches moins consommatrices d'énergie comme l'acquisition de données et la communication intra-grappe. De plus, les membres de chaque grappe sont, en général, adjacents et détectent des données semblables, qui sont agrégées par le CH, limitant de ce fait la quantité de données qui doit être envoyée à la station de base. Par ailleurs, la procédure de formation de grappes est seulement exécutée à l'activation du système, tandis que la reconfiguration des CHs est aussi longtemps que possible retardée.

Les taches principales de l'algorithme que nous avons proposé sont comme suit :

- Formation de grappes.
- Election d'un CH pour chaque grappe.
- Création d'un modèle de communication sans collision dans chaque grappe.

La procédure de formation de grappes se déroule en 5 étapes :

1. Former le groupe de voisinage $GV_i(1)$ de chaque nœud i. C'est à dire l'ensemble des nœuds qui sont dans sa portée de transmission.

$$GV_i = \left\{ j \in V / d(i,j) \prec tx_{range}(i) \right\} \quad (1)$$

2. Calculer les classes d'équivalence de chaque nœud *i*. Une classe d'équivalence EC_i (2) du nœud *i* est l'ensemble des éléments de son groupe de voisinage (GV_i), à l'exception des nœuds qui ne sont pas voisins entre eux ; c'est-à-dire le nœud *j* appartient à EC_i, si pour tout *k* appartient à EC_i, alors *j* et *k* sont voisins immédiats.

$$EC_i = \left\{ j \in GV_i / \forall k \in EC_i \Rightarrow j \in GV_k \right\} \quad (2)$$

3. Calculer le poids combiné W_{EC}, pour toute classe d'équivalence *EC* de chaque nœud. W_{EC} défini comme suit :

$$W_{EC} = \alpha \left| |EC| - \chi \right| + \frac{2\beta}{\left(|EC|^2 - |EC| \right)} \sum_{j,k \in EC} d(j,k) + \gamma \sum_{j \in EC} 1 / C_e(j) \quad (3)$$

Tels que α, β et γ sont des poids, $|EC|$ est la taille de la classe d'équivalence *EC*, χ est le seuil prédéfini limitant le nombre de nœuds qu'un CH peut idéalement gérer, $d(j,k)$ est la distance entre les deux nœuds *j* et *k* et $C_e(j)$ est le niveau d'énergie du nœud *j*.

4. Choisir la classe d'équivalence *EC* ayant le plus petit poids W_{EC} comme grappe.
5. Répéter les étapes 2 – 4 pour les nœuds restant, c'est-à-dire les nœuds qui n'appartiennent à aucune grappe.

Dans cette procédure de formation de grappes, nous pouvons constater que la métrique principale définissant le choix des grappes est W_{EC} (3). En effet, cette métrique est constituée de trois composantes qui reflètent : *(i)* la taille de la grappe, *(ii)* la distance entre les nœuds ; *(iii)* le niveau d'énergie de chaque nœud. La première composante contribue principalement à équilibrer la taille de chaque grappe afin de permettre aux CHs de gérer un nombre limité de nœuds qui tend vers un seuil prédéfini χ. En d'autres termes assurer un bon fonctionnement du modèle de communication intra-grappe. La deuxième composante est liée à la consommation d'énergie. Cette composante permet de mettre les nœuds adjacents dans une même grappe afin de minimiser l'énergie consommée due à la communication intra-grappe. De plus, elle permet de réduire la quantité de données à transmettre à la station de base car, en général, les nœuds de même proximité détectent des données semblables. La troisième composante mesure le niveau d'énergie des nœuds qui dépend aussi bien de la configuration initiale des nœuds que de l'énergie dépensée selon le trafic du réseau et la puissance de transmission. Cette dernière composante permet de distribuer les nœuds qui n'ont pas assez d'énergie dans différentes grappes afin de prolonger leurs durées de vie.

Ainsi, le RCSF est divisé en grappes locales, équilibrées et non recouvertes. Chaque nœud appartient à une seule et unique grappe. Chacune des grappes sera représentée par un CH, qui est élu parmi l'ensemble des nœuds de sa grappe. De ce fait, les nœuds ordinaires communiquent avec d'autres nœuds uniquement par le biais du CH de sa grappe. D'ailleurs, le CH est le nœud qui maximise le niveau d'énergie dans l'ensemble des nœuds de la grappe. A la différence d'un nœud simple, un nœud CH effectue des fonctions qui consomment beaucoup plus d'énergie. Il est donc inévitable de réélire périodiquement un CH afin de distribuer la consommation d'énergie sur l'ensemble des nœuds de la grappe et d'éviter ainsi la destruction rapide de certains nœuds. Ainsi,

chaque CH calcule périodiquement la moyenne d'énergie disponible de tous les nœuds de sa grappe. Si son niveau d'énergie actuel est en dessous de la valeur moyenne, alors le nœud ayant un maximum de réserve d'énergie est indiqué comme un nouveau CH de la grappe. Par conséquent, la réélection des CHs est le plus longtemps possible retardée.

La création d'un modèle de communication intra-grappe sans collision est la dernière étape liée à la phase de formation de grappes. Dans le but d'éviter les collisions entre les nœuds de même grappe, l'algorithme proposé utilise la technique de multiplexage temporel (TDMA) comme méthode d'accès au médium. Chaque nœud utilise la totalité de la bande passante allouée par le système de transmission durant son slot de temps. En fait, chaque CH agit comme un centre de commande local pour coordonner les transmissions des données dans sa grappe. Il crée un modèle de communication en se basant sur TDMA, ensuite il transmet ce modèle aux nœuds de sa grappe. Etant donné que chaque nœud connaît d'avance le slot de temps qu'il va occuper, cela permet alors au nœud de passer à l'état "endormi " durant les slots inactifs. Ainsi, la perte d'énergie due aux états de sur-écoute (overhearing) et d'écoute-passive (idle) est évitée.

4. Evaluation des performances et simulations

Dans cette section, nous étudierons les performances de l'algorithme de formation de grappes que nous avons proposé. Par la suite, nous comparerons cette proposition avec LEACH en terme de latence et de distribution des CHs dans l'espace.

Nous avons effectué plusieurs simulations de l'algorithme de formation de grappes en plaçant N nœuds dans une grille de *100m*100m* et nous avons mesuré certaines propriétés de l'algorithme. Dans le but d'évaluer cet algorithme, nous avons identifié 3 paramètres : *(i)* le nombre de grappes créées ; *(ii)* la connectivité ; et *(iii)* la latence. La connectivité est définie par la probabilité qu'un nœud est accessible par n'importe quel autre nœud, tandis que la latence représente la durée de temps qu'un nœud doit attendre pour accéder aux ressources partagées (canal, trame TDMA, …) de sa grappe. Ces trois paramètres sont étudiés en changeant le nombre de nœuds N dans le système, la portée de transmission et le seuil de la taille des grappes χ. Etant donné l'importance d'avoir des grappes de taille proche de la valeur de χ, le poids α associé à ce paramètre dans l'équation (3) a été choisi avec une valeur élevée. En revanche, les autres poids correspondant à la distance et à la puissance de batterie ont été utilisés avec des valeurs moins importantes. Les valeurs des poids utilisées dans les simulations sont $\alpha=0.5$, $\beta=0.25$ et $\gamma=0.25$. Notons que ces valeurs sont arbitraires et devraient être ajustées selon les conditions du système. Par ailleurs, pour ces premières simulations, N prend trois valeurs différentes qui sont 25, 50 et 100 nœuds.

La figure 3 montre le nombre de grappes créées en fonction de la valeur du seuil χ lorsque la portée de transmission de chaque nœud est égale à $tx_{rang}=30m$. Nous remarquons clairement que le nombre de grappes diminue quand la valeur de χ augmente. Ceci est prévisible, car si on incrémente la taille des grappes, alors l'algorithme de formation de grappes va automatiquement créer un nombre réduit de grappes. En effet, le nombre de grappes créées est proche de la valeur optimale égale à (N/χ). De plus ce nombre de grappes est plus élevé quand le nombre de nœuds est plus grand.

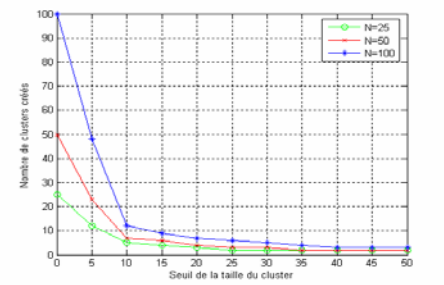


Figure 3. *Nombre de grappes créées, tx_{rang}=30m.*

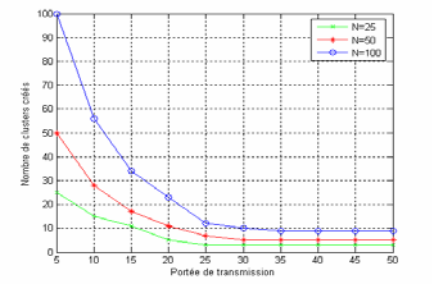


Figure 4. *Nombre de grappes créées, χ=10.*

Les résultats de la figure 4 représentent le nombre de grappes créées en respectant la variation de la portée de transmission avec χ=10. Nous observons que ce nombre diminue avec l'augmentation de la portée de transmission. Cela est dû au fait qu'un nœud couvrira un plus grand domaine avec une portée plus étendue de transmission, conduisant ainsi à augmenter la taille de son groupe de voisinage et celle de ses classes d'équivalence. Par conséquent, l'utilisation d'une portée de transmission assez élevée implique l'obtention de grappes ayant des tailles qui atteignent à peu près le seuil χ. A partir des résultats de la figure 4, nous pouvons noter que le nombre de grappes créées s'approche à N/χ lorsque la portée de transmission est plus grande que 25m.

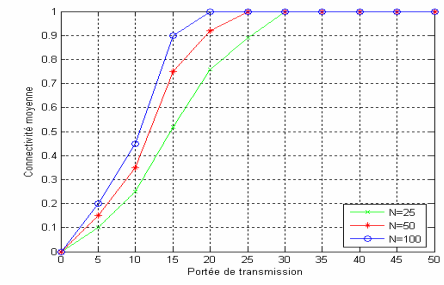


Figure 5. *Connectivité moyenne, χ=10.*

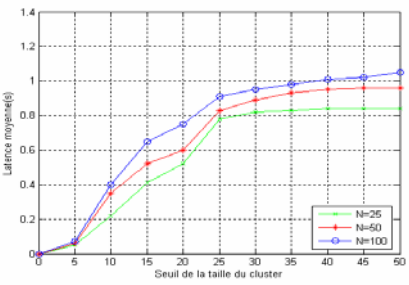


Figure 6. *Latence moyenne, tx_{rang}=30m.*

Nous allons maintenant étudier l'impact de la portée de transmission sur la connectivité des nœuds. Nous supposons un modèle de réseau idéal où un lien est établi entre deux nœuds quelconques se trouvant dans la portée de transmission de l'autre. Un chemin peut toujours être trouvé entre deux nœuds reliés par une chaîne de liens. Plusieurs déploiements aléatoires de *N* nœuds sont examinés pour chaque portée de transmission. La figure 5 représente la connectivité moyenne de ces déploiements. Ce schéma démontre qu'un graphe connexe peut être obtenu en utilisant une portée de transmission plus élevée. A partir de ces résultats, nous déduisons que la portée de

transmission garantissant une connectivité totale du réseau (tous les noeuds sont connectés) doit être de : *(i) 20m* dans le cas où *N* est égal à *100* ; *(ii) 30 m* dans le cas où *N* est égal à *25*.

La figure 6 montre la latence moyenne par rapport au seuil χ, avec une portée de transmission (tx_{rang}) égale à *30m*. Nous observons que la latence moyenne augmente avec l'incrémentation du seuil χ. D'une part, une petite valeur de χ mène à la création de grappes de petites tailles, ce qui cause une mauvaise utilisation des ressources. D'autre part, une grande valeur de χ conduit à diviser le réseau en grappes de grandes tailles, mais cela augmente la latence et le surcoût (overhead) de communication.

Nous allons maintenant comparer notre algorithme de formation de grappes avec LEACH en termes de la distribution des CHs dans l'espace et de la latence maximale. Nous évaluons la distribution des CHs par la moyenne des distances entre les nœuds et leurs CHs. Nous fixons cette fois le nombre de nœuds (*N) à 100*, et nous supposons que chaque nœud peut communiquer avec n'importe quel autre nœud dans le secteur du réseau *100m*100m*.

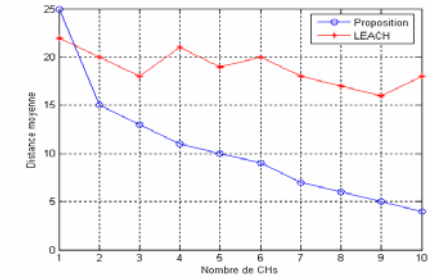


Figure 7. *Distribution des CHs.*

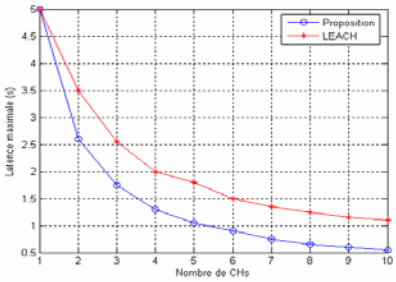


Figure 8. *Latence maximale.*

La figure 7 montre la distance moyenne entre les nœuds et leurs CHs en fonction du nombre de ces derniers. L'algorithme de formation de grappes proposé dépasse clairement LEACH lorsque le nombre de CHs augmente, car le choix aléatoire des CHs dans LEACH peut mener à une distribution irrégulière des nœuds élus. En effet, dans le cas où les CHs sont voisins ou situés à la frontière du réseau, les nœuds éloignés des CHs nécessitent des puissances d'émission très élevées pour la communication intra-grappe. Notre algorithme résout ce problème en effectuant une distribution équitable des CHs dans les différentes régions du RCSF. Une distribution régulière des CHs permettrait, évidemment, d'économiser l'énergie des nœuds, alors qu'une distribution irrégulière des CHs mènerait à une dissipation inutile d'énergie.

La figure 8 présente la latence maximale en fonction du nombre de CHs. Nous constatons clairement que notre algorithme affiche de meilleures performances que LEACH. Cela est dû au fait que cet algorithme distribue équitablement les nœuds dans les grappes, tandis que dans LEACH, certaines grappes peuvent être surchargées, et d'autres grappes contiennent peu de nœuds.

5. Conclusion

Dans cet article, nous avons présenté un nouvel algorithme de formation de grappes qui combine efficacement différents paramètres (la taille de la grappe, la puissance de transmission et l'énergie des nœuds) pour organiser les nœuds (capteurs) dans des grappes équilibrées et non recouvertes. Cet algorithme assigne différents poids à ces paramètres suivant leur importance dans la composition de la grappe. Les avantages associés à ce processus de formation de grappes sont : *(i)* le groupement des nœuds adjacents dans une même grappe permet de minimiser la consommation d'énergie lors de la communication intra-grappe et de favoriser l'agrégation ainsi que la compression des données ; *(ii)* le contrôle de la taille des grappes permet aux CHs de gérer efficacement les nœuds de leurs groupes et garantit un bon fonctionnement du système ; *(iii)* la non périodicité de l'algorithme permet de réduire les surcoûts de calcul et de communication. Cet algorithme réalise également une distribution assez uniforme de CHs à travers le réseau de capteurs. Les résultats de simulation ont montré que le nombre de grappes créées diminue avec l'augmentation du seuil χ et de la portée de transmission. Cependant, la latence moyenne augmente lorsque le seuil χ augmente. Ainsi, un bon fonctionnement du système peut être réalisé en limitant ou en optimisant la taille de chaque grappe. Les résultats de simulation ont également montré que notre algorithme a de meilleures performances que LEACH.

6. Bibliographies

I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Network, vol.38, no.4, pp.393-422*, 2002.

S. Basagni, I. Chlamtac and A. Fargo, "A Generalized Clustering Algorithm for peer-to-peer Networks," *Proceeding of Workshop on Algorithmic Aspects of communication (satellite workshop of ICALP)*, July 1997.

T. J. Kwon and M. Gerla, "Clustering with Power Control," *proceedings of IEEE MILCOM'99, Atlantic City, NJ*, November 1999.

C. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," *In IEEE Journal on Selected Area in Communications, volume 15, pp. 1265-1275*, September 1997.

S. Basagni, "Distributed Clustering for Ad Hoc Networks," International Symposium of Parallel Architectures, Algorithms and Networks (I-SPAN'99), Fremantle-Australia, June, 1999.

W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application Specific Protocol Architecture for Wireless Micro Sensor Networks," IEEE Transactions on wireless Communications, vol. 1, no. 4, (pp. 660-670), October 2002.

O. Younis, S. Fahmy, "Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-Efficient Approach," *Proceedings of IEEE INFOCOM 2004, Hong Kong*, March 2004.

A. Harter and A. Hopper, "A New Location Technique for the Active Office," IEEE Personal Communication, vol. 4, No. 5, (pp. 42-47), October 1997.

N. B. Priyantha, A. Chakraborty, H. Balakrishnan, "The Cricket Location Support System," Proceedings of 6th ACM Mobicom, Boston, MA, August 2000.

A. Savvides, H. Park and M. B. Srivastava, "The n-hop Multilateration Primitive for Node Localization Problems," Proceedings of Mobile Networks and Applications, 8, (pp. 443-451), 2003.

Ubisense website, http://www.ubisense.net

Gestion de vie de réseaux overlays pour les réseaux ambiants

Meng Song
France Telecom, R&D,France
2, rue, Pierre Marzin, 22303, Lannion
meng.song@francetelecom.com

Bertrand Mathieu
France Telecom, R&D,France
2, rue, Pierre Marzin, 22303, Lannion
Bertrand2.mathieu@francetelecom.com

Noëmie Simoni
ENST, Paris, France,
46, rue Barrault - 75013 Paris
simoni@enst.fr

RÉSUMÉ : *Les réseaux ambiants sont de plus en plus étudiés, notamment sur les aspects mobilité, la gestion des interfaces radio, la sécurité et la prise en compte du contexte car l'objectif final reste toujours de fournir un service de qualité. Les réseaux d'accès étant variés (Wifi, Wimax, UMTS, ad hoc…) et les terminaux de plus en plus nombreux (PDA, PCs, SmartPhone…), il est apparu nécessaire de fournir un contenu adapté au contexte des utilisateurs. Ce papier présente une solution basée sur l'utilisation de réseaux overlays spécifique au service. Ainsi un réseau overlay est créé, pour permettre aux services de fournir un contenu adapté, non seulement en tenant compte des caractéristiques des réseaux physiques sous-jacents mais aussi en fonction des caractéristiques du service. Par exemple, le réseau overlay peut intégrer des nœuds effectuant des traitements spécifiques (transcodage, cache, synchronisation de flux…). Cependant, une fois créé, le réseau overlay n'est pas figé. En fonction des utilisateurs se connectant ou se déconnectant au service, le réseau overlay doit pouvoir être étendu (rajout de nœuds offrant le traitement adéquat) ou diminué (suppression de nœuds devenus inutiles). La maintenance et la gestion de vie du réseau overlay deviennent donc un problème crucial. Dans ce papier, une solution de type "soft-state" est présentée pour la gestion de vie du réseau overlay. Cette solution repose sur la surveillance périodique de l'état des nœuds voisins et un échange d'informations qui permettent de décider des actions à entreprendre sur le réseau overlay.*

*MOTS CLÉS**:** réseau overlay, gestion de vie du réseau overlay, réseau ambiant.*

1 Introduction

Depuis le déploiement de réseaux d'accès variés (Wifi, Wimax, UMTS, ad hoc…) et l'arrivée de terminaux de plus en plus nombreux (PDA, PCs, SmartPhone…), un nouveau besoin est apparu : celui de fournir un contenu adapté au contexte des utilisateurs. Les utilisateurs possédant divers terminaux et étant mobiles, la notion de réseau ambiant a émergé ces dernières années. Ils sont de plus en plus étudiés, notamment sur les aspects mobilité, la gestion des interfaces radio, la sécurité mais très peu traitent de la problématique de l'adaptation de contenu pour ces réseaux ambiants. Ce papier présente une solution basée sur l'utilisation de réseaux overlays spécifiques au service, dénommée SSON (Service Specific Overlay Network), supporté par une architecture de réseau ambiant, pour permettre cette adaptation. Ainsi un réseau overlay est créé, pour permettre aux services de fournir un contenu adapté, non seulement en tenant compte des caractéristiques des réseaux physiques sous-jacents mais aussi en fonction des caractéristiques du service. Par exemple, des fonctionnalités de traitement applicatives spécifiques telles que le transcodage, la gestion de cache, la synchronisation de flux, sont intégrés dans le réseau overlay. Ce réseau overlay est crée par service et sur demande mais la topologie de ce réseau peut évoluer en fonction des utilisateurs se connectant ou se déconnectant au service. Ainsi il doit pouvoir être étendu (rajout de nœuds offrant le traitement adéquat) ou diminué (suppression de nœuds devenus inutiles) en fonction du contexte. La maintenance et la gestion de vie du réseau overlay, depuis la phase de création jusqu'à la phase de destruction en passant par les phases d'adaptation, deviennent donc un problème crucial à gérer. Dans ce papier, une solution de type "soft-state" est présentée pour la gestion de vie du réseau overlay. Cette solution repose sur la surveillance périodique de l'état des nœuds voisins et un échange d'informations qui permettent de décider des actions à entreprendre sur le réseau overlay.

2 Présentation des réseaux ambiants

Le projet européen IST "Ambient Networks" vise à étudier les réseaux ambiants selon différentes perspectives; composition de réseaux, gestion, interfaces radio, mobilité, fourniture de contenus adaptés à l'utilisation de réseaux overlays, connaissance du contexte,… . Ce paragraphe regroupe quelques idées majeures issues de ce projet et qui définissent le contexte des réseaux ambiants dans lequel nous proposons notre solution.

Les réseaux ambiants peuvent être définis comme une architecture ouverte d'interconnexion de réseaux, utilisant des infrastructures différentes, qui se composent dynamiquement. Ils peuvent être illustrés par exemple par le scénario, dans lequel les réseaux IP mobiles (WLAN, Wimax …), les réseaux radio mobiles (GPRS, UMTS...) et les réseaux personnels (PAN) sont composés pour fournir une

plateforme ouverte permettant le développement de services. Dans le cadre du projet "Ambiant networks", une interface commune (ASI pour Ambiant Service interface) a été définie pour permettre un accès au réseau ambiant. Ainsi le même type de service peut être fourni par plus d'un fournisseur. D'ailleurs, le modèle de communication de type Peer-to-Peer (P2P) a été défini dans le cadre du projet permettant aux utilisateurs de jouer les deux rôles (client/serveur) à la fois.

Selon le contexte de l'utilisateur, ceci est d'autant plus vrai que la mobilité de l'utilisateur et la communication sont omniprésentes. En effet, les réseaux ambiants intégrant les réseaux mobiles, des réseaux d'accès spontanés et des terminaux variés (PDA, ordinateur portable...), ils doivent gérer le passage d'un réseau d'accès à un autre, d'un terminal à un autre pour fournir le service demandé par l'utilisateur, quelque soit sa localisation et s'adapter à son terminal. Ainsi, il est possible que plusieurs traitements applicatifs sur le contenu soient nécessaires pendant une même session de service. Des services de réseau ambiants se composent donc de plusieurs composants de service intermédiaires, par lesquels, les flux passeront pour être y "traités".

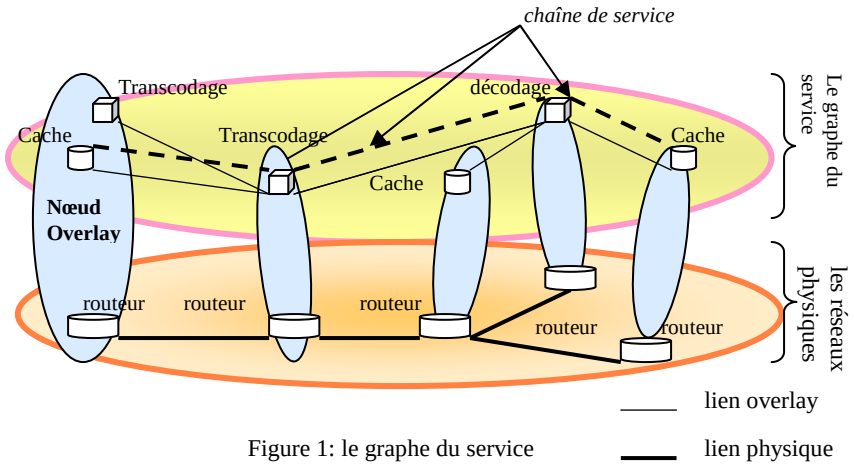


Figure 1: le graphe du service

Dans la figure 1, les blocs cube et les cylindres représentent les composants de service tels que le cache, transcodage, encodage etc… Ils sont fournis par des nœuds spécifiques (nœuds overlay) qui sont connectés entre eux par des liens virtuels (liens overlays), représentés par des liens unis fins sur la figure 1. Une "*chaîne de service*", représentée par les lignes pointillées dans la figure 1 est l'ensemble des composants de service...Les traitements d'adaptation des flux sont définis par l'ordre des composants de service traversés. Grâce au mapping de la chaîne de service au réseau overlay, on a obtient un graphe de service, composé de nœuds overlay et de liens de connexion. Notamment, la topologie du graphe de service et les nœuds overlay sous-jacent sont adaptés aux réseaux physiques de manière transparente.

Le projet Ambiant Networks a défini une architecture SMART (Smart Multimédia Architecture Routing and Transport) permettant l'insertion de modules

de traitements applicatifs (*transcodage, cache, synchronisation*) dans le chemin de bout-en-bout. Dans la figure 1, les lignes de couleur différente distinguent les différents types de réseaux overlay, chacun correspondant à un contexte spécifique d'un service. Cette notion de réseau overlay spécifique au service est dénommé SSON pour Service Spécifique Overlay Networks et est décrite dans le paragraphe suivant, après une description de l'architecture SMART.

3 Concept SMART & Architecture

La plupart des nouveaux services de réseau ambiant sont des services multimédias. La définition de l'architecture SMART (*Smart Multimedia Architecture for Routing and Transport*)[3][4] vise à optimiser les performances de transport pour les flux de service du réseaux ambiants, notamment, lorsqu'il s'agit de réseaux ou de terminaux qui ont des contraintes spécifiques telles qu'une faible bande passante, un petit écran pour un PDA etc. En général, SMART vise à optimiser le transport, en profitant de services intermédiaires inclus dans le réseau. Ces services intermédiaires, appelés *MediaPorts (MP)*, sont des entités tierces qui fournissent les traitements applicatifs permettant de délivrer un service adapté aux contextes. Ces MPs peuvent offrir des fonctionnalités de cache, de transcodage, de synchronisation ou de routage applicatif.

Pour réaliser le concept SMART, le projet "Ambient Network" a choisi d'utiliser le concept de réseau overlay. Un réseau overlay consiste en une couche virtuelle au dessus des réseaux physiques, qui implémente ses propres mécanismes de contrôle et définit et maintient un protocole de routage au niveau overlay, indépendant du routage physique sous-jacent. Cette solution permet de faire évoluer le réseau overlay, indépendamment des réseaux physiques, permet d'optimiser le routage en s'adaptant en cas de problèmes dans les réseaux physiques, tels que les congestions de réseau, l'arrivée ou départ de nœuds mobiles….Des études ont déjà démontré son intérêt. Dans RON[5] un réseau overlay est créé pour rendre le transport plus fiable et résilient, ce qui permet de compenser rapidement en cas de problèmes (panne, réseau bloqué, ou surchargé etc). Dans QRON[2], qui est une extension de RON, l'objectif est de fournir une architecture de base de réseau overlay pour améliorer la qualité du réseau Internet, principalement en utilisant un routage logique, évitant les nœuds critiques des réseaux physiques. Dans MBONE [6], le réseau overlay est utilisé pour transporter des paquets multicast tout en optimisant le transport et l'efficacité du réseau pour le multicast. XBone, lui, fournit à l'utilisateur une interface graphique, qui permet de créer un réseau overlay IP au-dessus du réseau IP [7]. L'objectif de ce type de réseau overlay est de permettre la construction d'un *testbed* aussi facile que possible.

Dans le cadre du projet Ambiant Networks, les motivations pour utiliser un réseau overlay sont les suivantes. L'établissement d'un réseau overlay est transparent aux réseaux physiques supportés. Le réseau overlay est virtuel, dans lequel les composants du service peuvent se trouver sur n'importe quel nœud physique. D'autre part, les nœuds sont reliés par les liens virtuels qui sont indépendants du réseau

physique. Un lien virtuel overlay peut donc traverser plusieurs réseaux physiques. Le réseau overlay fournit une plateforme ouverte pour l'accès aux services. Ses caractéristiques de *plug-and-play* lui permettent de supporter des services à valeur-ajoutée, d'accepter des services d'adaptation ou d'ajouter des routeurs intermédiaires. En cas de disparité entre les caractéristiques de service et les requis particuliers du côté de l'utilisateur (en terme de capacités du terminal, des préférences d'utilisateur…), il est possible de rajouter les composants du service supplémentaires adéquats dans le réseau overlays et ils sont insérés dans la chaîne de service. De plus, il est également possible d'ajouter de nouvelles routes pour étendre le service à de nouveaux domaines de réseau.

Pour prendre en compte les caractéristiques (décrites ci-dessus) que doit permettre un réseau ambiant, le projet "Ambient Network" a défini une architecture pour pouvoir rendre les services dans les réseaux ambiants. La figure 2 présente cette architecture à trois niveaux : le niveau de service, le niveau de contrôle de réseaux (management), et le niveau de transport des données. La couche service permet à un client ou fournisseur de service d'accéder aux réseaux ambiants afin de créer un SSON, puis de lancer une session de service et de gérer les clients de son service sur son SSON. Ces messages sont spécifiés suivant l'interface ASI (Ambient service interface). Pour satisfaire les requis des utilisateurs, et pour rendre transparent le service fourni, le projet "Ambient Network" s'appuie sur un espace de contrôle ambiant, l'ACS (*Ambient Control Space*)[1]. Cet ACS est composé de plusieurs entités de contrôle telles que la gestion de la sécurité, de la mobilité et du contexte, de la fourniture de contenus adapté et aussi l'espace de contrôle des réseaux overlay, OM (Overlay Management). Les nœuds overlays se retrouvent dans le niveau de transport des données.

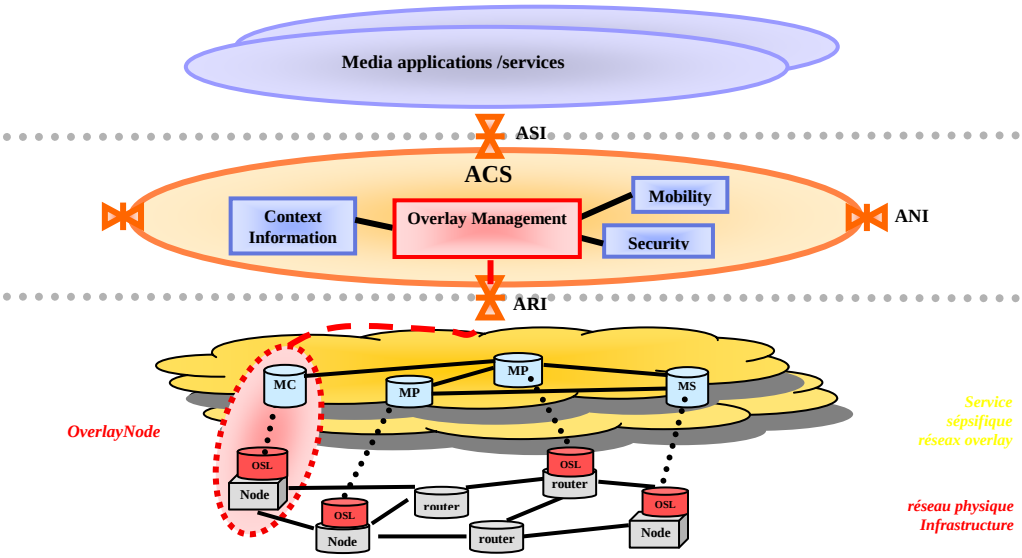


Figure 2: Le concept et l'architecture du SMART

L'OM (Overlay Management) est le domaine fonctionnel de l'ACS qui contrôle les réseaux overlays spécifiques au service (SSON). Il contrôle la création des SSONs à la demande des utilisateurs (par exemple des fournisseurs de service, des clients, etc...). L'OM contrôle aussi la réorganisation ou l'adaptation de SSONs existants en fonction des changements de contexte des réseaux sous-jacents (connectivité, composition, QoS, mobilité, fonctions de traitement de médias, etc..), des changements de contexte de l'utilisateur (par exemple, localisation, profil utilisateur, possibilités de terminal) ou des changements de polices du service.

Les ONodes (Overlays Nodes) sont les nœuds du réseau overlay spécifique (SSON) et sont localisés dans le réseau, ou dans le réseau d'accès du terminal. Ils permettent la création du réseau overlay pour les services en réservant les ressources nécessaires et hébergent les MPs, réalisant des fonctions de routage des données ou des traitements applicatifs réalisés dans le réseau overlay (cache, adaptation, synchronisation, routage…). Un ONode peut héberger plusieurs MPs et donc offrir plusieurs fonctions d'adaptation. Dans l'architecture SMART, en plus des MPs, deux autres modules ont été définies : *1) Le MC (Media client*) qui est le point d'entrée des données du service sur le terminal. C'est donc le point de sortie des flux de données du service du SSON, 2) *Le MS (Media Server)* qui est l'opposé du MC. Il est le point de sortie des données du service du terminal et donc le point d'entrée des données du service dans le SSON.

Le OSL (overlay support layer) est la couche de connexion entre les nœuds overlay. Dès qu'un paquet arrive à un nœud overlay, la couche OSL va détecter l'identifiant de son SSON participant, l'adresse de la source et de la destination. Selon la table de routage spécifique du SSON, l'OSL renvoyer ce paquet soit au ses propres MPs, soit au nœud overlay suivant.

4 Gestion de Vie du SSON

Puisqu'un SSON est créé pour un service donné, la durée de vie d'un tel service est variable (de la durée de quelques minutes, comme un service conversationnel de téléphone, jusqu'à des mois ou même l'année, par exemple, pour un service de diffusion vidéo). La durée de vie pour un SSON peut être limitée, mais les questions de sa gestion se posent bien évidemment dans tous les cas. Comment créer un SSON ? Comment modifier un réseau overlay existant pour pouvoir garantir une certaine qualité de service ? Comment détruire le réseau overlay à la fin du service? Dans les paragraphes suivants, nous définissons des déclencheurs (triggers) qui sont responsables de gérer les changements de l'état du réseau overlay, et nous définissons le mécanisme de contrôle « soft-state » pour le réseau ambiant. Enfin, un scénario présente le processus d'une approche de destruction de SSON (teardown).

4.1 EVENEMENT DE GESTION DE VIE DU SSON

Puisque le SSON peut être créé, modifié, et détruit, il a été défini certains événements qui déclenchent la modification d'un SSON :un SSON peut être détruit sur demande d'utilisateurs, ou de service provider(s). Dans les deux cas, les points terminaux peuvent envoyer un message d'arrêt du service (i.e. message d'*EndSession*) au point de contrôle du réseau overlay, afin qu'il puisse modifier les tables de routage appropriées et adapter le SSON. (Voir dans la figure 3, ces sont des triggers externe du AN.). La reconfiguration ou l'arrêt d'un SSON peut également être déclenchée par l'information de contexte. L'information reliée à la QoS, par exemple, doit être prise en compte. Si un nouveau service multimédia utilise beaucoup de ressources (bande passante) pour transporter ses données, alors la QoS pour certain *MediaClient* peut se dégrader, ou même la qualité du service peut éventuellement chuter pour tous les utilisateurs de ce SSON. La mobilité est une autre information qui doit être prise en compte. Par exemple si un utilisateur mobile n'a pas obtenu les ressources suffisantes pour continuer son service dans une nouvelle zone de réseau, alors le SSON doit être adapté ou détruit. D'autres informations de contexte telles que l'information de facturation, de sécurité de réseau

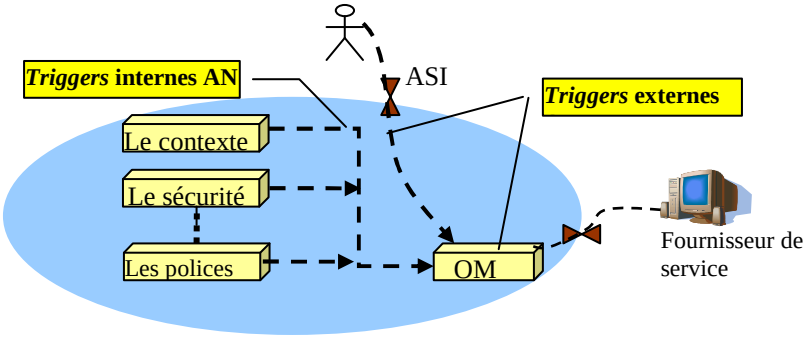


Figure3: Les triggers pour la gestion de la vie du SSON

d'accès, de localisation peuvent aussi déclencher les triggers (voir dans le schéma 3, les triggers internes du AN) pour modifier la vie du réseau overlay. Cette situation est expliquée plus loin dans cet article (voir §4.3).

La figure 3 indique les triggers pour la gestion de la vie du réseau overlay. Les triggers externes sont déclenchés soit par le fournisseur de service soit par l'utilisateur final. Tandis que les triggers internes proviennent des entités de contrôle des réseaux ambiants, telles que la gestion du contexte, la gestion des politiques, la gestion de la sécurité… Il y a également des mécanismes de contrôle du réseau overlay qui gèrent la gestion de vie de SSON. Dans ce cas, chaque nœud du réseau overlay contrôle ses propres informations et celles de ses voisins pour reconfigurer les nœuds du SSON, ou les détruire. Ainsi le système est capable de détecter l'état du réseau et de prendre les actions adéquates. Par exemple, si l'OM détecte des

défaillances de certains composants du service, le SSON peut être adapté ou terminé par l'OM. Nous caractérisons ce mécanisme de déclenchement initié par le réseau overlay, la gestion de la vie de type *"soft-state"*.

4.2 GESTION DE VIE DE TYPE "SOFT-STATE"

La gestion de vie de type "soft-state" permet à chaque nœud du SSON de surveiller son état et également de conserver l'information sur les capacités de ses voisins dans une base de donnée locale. En conséquence, un nœud envoie périodiquement un message à ses voisins pour détecter les états en temps-réel et mettre à jour sa base de données. Ces mesures et la mise à jour de la base de données sont contrôlées par plusieurs durées d'expiration ("timers") qui indiquent la durée de vie prévue du SSON, le temps d'attente avant l'émission des messages pour renouveler l'information de ses voisins... Les messages entre un nœud overlay et ses voisins sont des messages de signalisation tels que *KeepAliveMessage* (KAM), *UpdateLifetimeMessage* (ULM) utilisés pour détecter et surveiller les états des ONodes ainsi que l'état du lien de connexion entre eux et leurs capacités disponibles. La méthode de la gestion de vie de type "soft-state" offre une solution flexible pour mettre à jour la base de donnée en échangeant l'information automatiquement et pour gérer la topologie du réseau overlay en tenant compte de l'information du contexte.

La gestion de vie de type "soft-state" se justifie par le fait que la qualité d'un chemin de transport fiable ne peut pas être assuré dans les réseaux ambiants. Ainsi les données transportées par des réseaux ambiants peuvent être perdues, dupliquées ou bloquées à un point quelconque dans le chemin de réseau, notamment dans les réseaux d'infrastructure spontanés tel que la PAN et réseau ad hoc, où leurs contraintes matérielles, comme la largeur de bande passante, sont limitées, et leurs connexions faibles peuvent dégrader la qualité de service offerte. D'ailleurs, la perte de messages de signalisation peut avoir un résultat défavorable et même déclencher la destruction du SSON. Dans ce cas, s'il n'y pas un mécanisme pour contrôler le cycle de vie de SSON, le problème ne peut pas être résolu sans une intervention de l'opérateur de réseau. Pour éviter les problèmes provoqués par la perte des messages de contrôle, lorsqu'on établi un nouveau SSON, le paramètre de la durée de la vie va être initialisé. Avant l'expiration de cette valeur, les nœuds du réseau overlay envoient périodiquement leur état vers leurs voisins et acquièrent également l'information des voisins dans le message de réponse. De cette façon, la durée de vie de SSON peut être vérifiée, modifiée, prolongée ou terminée.

Cette gestion de vie de type "soft-state" s'adapte bien à l'infrastructure des réseaux ambiants. En effet, du point de vue de l'utilisateur terminal du SSON (un utilisateur ou un fournisseur de service), ils offrent des profils de service, contenant la durée prévue pendant laquelle le service sera opérationnel. Selon cette information du profil de service, l'OM peut créer un nouveau SSON ou modifier un SSON existant pour permettre la nouvelle demande. À la fin d'un service, le fournisseur de service peut supprimer le SSON à moins que sa durée de vie ait été prolongée, sur demande de l'utilisateur ou du fournisseur de service. De cette façon, les ressources

réservées pour un SSON donné ont pu être récupérées par le système, et réutilisées pour un autre SSON (pour un autre service). Du point de vue du système, le fait que le SSON peut autocontrôler sa durée de la vie permet de réutiliser les ressources de manière la plus efficace possible. De plus, chaque SSON est associé à une valeur d'expiration (timeout) pour l'ensemble des nœuds du SSON et il comporte aussi des valeurs différentes pour chacun des nœuds à l'intérieur du SSON. Si la durée de vie d'un nœud du SSON a expiré, ce dernier sera retiré du SSON. Ce mécanisme assure que les ressources d'un ONode peuvent être contrôlées et libérées même en cas de perte de messages de contrôle.

4.3 GESTION DE VIE DU RESEAU OVERLAY

Le cycle de vie de SSON est divisé en trois étapes : l'établissement, l'adaptation et la destruction.

Etablissement

Après réception d'une demande de service de réseau ambiant par l'interface ASI, l'OM décide de créer un SSON (s'il n'existe pas) afin de transporter les données multimédia. Pour cela, deux étapes sont nécessaires. La première étape consiste à établir la composition du service et à mettre en communication tous les nœuds overlay pour pourvoir construire la topologie du réseau. La deuxième étape est relative au transport des données multimédias par la couche support du réseau overlay - OSL (Overlay Support Layer). Un protocole de routage spécifique pour ce SSON peut être instancié et initialisé et la table de routage mise-à-jour dans la couche OSL. Par exemple un protocole de routage de type "source routing" ou un protocoles de Peer to Peer comme CAN, Chord, Tapestry/Pastry, peuvent être utilisés tous en respectant les caractéristiques spécifiques du réseau, et les spécificités de service. Les études relatives à la composition du service, à la localisation des services intermédiaires, au routage overlay et aux traitements eux-mêmes sont prises en compte dans le projet "Ambiant network" mais ne sont pas abordés dans cet article, qui se focalise sur le cycle de vie du SSON, et propose une solution de "soft-state" pour rendre le contrôle du cycle de vie de SSON plus flexible et dynamique.

Pour la méthode de contrôle de type "soft-state", chaque SSON est associé à une durée d'expiration, qui précise la durée de présence d'un SSON pour fournir le service de réseau ambiant. Il est possible d'obtenir cette valeur d'expiration de manière explicite ou implicite. Par exemple : dans un premier cas, un des participants du service de conférence peut réserver une salle pendant une durée limitée ; dans un deuxième cas, la durée de vie peut être obtenue par l'information de contexte ou indiqué par la préférence de l'utilisateur. La durée d'expiration peut être initialisée par défaut à une valeur minimum à l'établissement du SSON. Cette

information sera stockée et transportée comme un des éléments des profils de service.

Dès qu'un réseau de SSON est créé, il est opérationnel jusqu'à ce que sa durée de vie soit expirée. Cette durée de vie du SSON n'est pas obligatoirement une valeur fixe, mais peut être modifiée si nécessaire. Par exemple, un réseau overlay pour un service de diffusion vidéo est opérationnel jusqu'à ce que le dernier client ait quitté le service. Pendant le processus d'établissement de SSON, l'ensemble de nœuds SSON peuvent annoncer un intervalle de durée à l'OM pour initialiser la durée de vie. Néanmoins, chaque nœud participant au SSON peut avoir une durée de vie différente. Certains clients peuvent rester plus longtemps dans un SSON que les autres. Ce peut être le cas d'un service qui refuse de continuer à offrir le service aux utilisateurs, qui n'ont pas payé. Chaque nœud dans SSON gère son propre état et en même temps, détecte et surveille l'état de ses voisins. Ces informations sont rafraichies périodiquement et dynamiquement.

Maintenance / adaptation

Pendant la durée de vie du SSON jusqu'à son expiration, la composition du réseau overlay, sa topologie et son modèle de communication (Client/server, peer to peer) peuvent être modifiés en tenant compte du changement du contexte ; ce processus est appelé adaptation du SSON. Les événements qui déclenchent l'adaptation de SSON peuvent être séparés en deux catégories. Dans le premier cas, on peut seulement reconfigurer le SSON, en modifiant les paramètres. C'est une méthode nommé "adaptation légère de SSON". Au contraire, il y une autre méthode, appelée "adaptation lourde de SSON" qui non seulement modifie les paramètres mais aussi ajoute ou enlève des nouveaux modules d'adaptation de service, des liens virtuels qui sont saturés ou définit de nouvelles routes dans le réseau SSON.

Plusieurs événements peuvent déclencher la fonction de reconfiguration de SSON : les changements du réseau physique (caractéristiques de QoS, etc..), les changements de la qualité de MPs, de MSs ou de MC (par exemple, la composition/décomposition des réseaux ambiants), les changements du contexte de l'utilisateur ou du service, les décisions interne de l'OM (politiques de gestion, durée d'expiration, etc), l'introduction de nouvelles fonctionnalités dans le SSON. Dans ces cas, il est possible qu'un MediaPort (cache, transcodeur...) doive être ajouter/supprimer dans un chemin de bout en bout de service pour réaliser un processus d'adaptation ou pour l'enlever. Dans ce cas, l'adaptation de SSON fera un changement de topologie en modifiant plusieurs paramètres du réseau, les nœuds, et sa valeur de vie.

Certains événements, comme l'échec d'un ONode ou une congestion du réseau peuvent déclencher uniquement une action de modification du routage dans le SSON. Ces événements ont pu provoquer une modification du routage dynamique et donc changer le chemin dans le réseau overlay [8]. Cette action est légère, de telle sorte qu'elle puisse rapidement réagir aux changements de réseau ou relatifs à

l'utilisateur. D'ailleurs, cette logique ne peut pas ajouter ou enlever des ONodes du SSON, elle décide uniquement quel chemin suivre parmi les ONodes dans le SSON.

Pour mettre à jour la durée de vie du SSON, l'OM envoie un message, de type "*SetLifeTime*", aux nœuds overlay via l'interface ORI (overlay ressource interface). Ainsi, les informations communes du SSON peuvent être mise à jour.

Dans le cas d'une gestion de type "*soft-state*", les nœuds overlay diffusent périodiquement leur propre information à leurs voisins, y compris leurs caractéristiques fonctionnelles (cache, transcodeur...), leurs capacités réseau et de traitement (puissance du traitement, largeur de bande, le délai du service, le délai du transport...) et leurs propriétés (prix, propriétaire...). Chaque émetteur peut recevoir les informations de son voisin dans un message de réponse. Bien évidemment, il faut trouver un compromis entre le but d'obtenir des informations des nœuds en temps réel et la réduction des messages de contrôle afin de ne pas saturer le réseau. Par exemple, dans un SSON associé à un service multicast, la topologie est de type arbre de diffusion. Chaque nœud envoie les messages de *KeepAlive* (KAM) à ses "nœuds fils", ce processus s'appelle "*prune*".

Arrêt/destruction

Un fournisseur de service peut terminer un SSON selon différentes situations : à la fin du service, lors de problèmes de transport qui n'ont pas pu être résolus, lorsqu'il ne reste plus d'utilisateur dans ce réseau, lors de congestion de réseau insoluble ou des problèmes de nœuds, en fonction d'information du contexte ; par exemple un manque du crédit utilisateur (en cas de servie payant) peut arrêter le service en cours….

Dans cette partie, trois cas différents pour la destruction du SSON ont été définis : la destruction sur commande où la destruction de SSON est commandée par les points terminaux du SSON ; la destruction sur *triggers* où l'expiration de la durée de vie du SSON inpose de le détruire, et la destruction de type *teardown* où la destruction du SSON est faite progressivement niveau par niveau, selon le mécanisme de gestion de vie de type "soft-state".

Les terminaux ou des informations de contexte peuvent commander la destruction d'un SSON. Dans ce cas, un message *EndSession* devrait être envoyé au module OM, qui transmettra ensuite un message de *SSONRelease* à partir du nœud de racine (MS) jusqu'aux nœuds appartenant au même SSON. Finalement, les ressources réservées à ce SSON seront libérées.

Dans le cas de la destruction sur *trigger*, chaque ONode envoie un message de type *KeepAlive* à ses nœuds voisins au moment où sa durée de vie expire. En fonction du service, le contrôle OM permet aux terminaux de SSON de rafraichir leur accès au service (extension de la durée de vie) ou de le terminer et de détruire le SSON pour leur accès.

Par rapport aux deux cas précédents, la destruction en *teardown* n'est commandée ni par l'utilisateur ni par l'information de contexte. Chaque nœud de SSON envoie le message de signalisation à ses voisins et attend les réponses, pour mettre à jour la durée d'expiration et pour maintenir l'état de ses voisins. Au cas où une durée de vie a expiré avant une mise à jour de la valeur et la prolongation de la durée de vie, le nœud sera supprimé de ses nœuds voisins. Ainsi niveau par niveau, le SSON sera entièrement détruit. Ce scénario peut être illustré dans la figure 4 et expliqué ci-dessous.

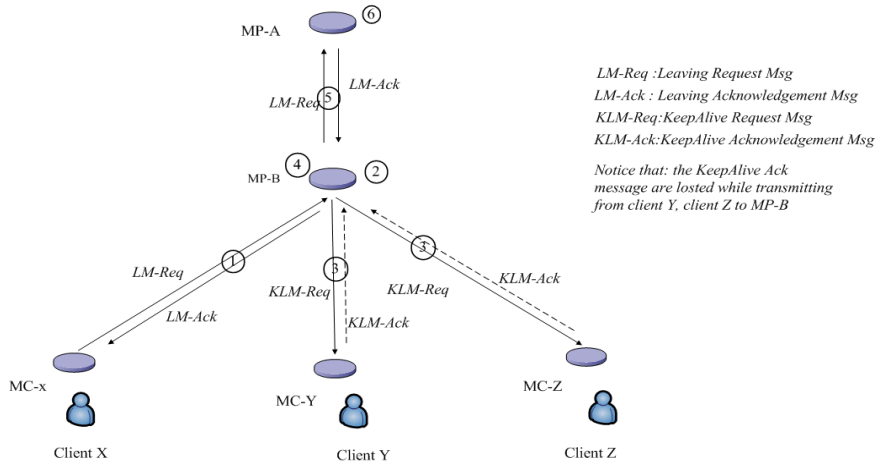


Figure 4: La destruction en *teardown*

Dans cet exemple, un SSON est composé de deux MP (*Mediaport)* tels que MP-A et MP-B, et de trois *clients*, chacun correspondant à un MC (*Mediaclient)*, soit MC-x, MC-y et MC-z.

1) Si le client X souhaite arrêter le service, il enverra *un msg de type LeavingRequest* à son ONode père (MP-B)

2) Sur réception de cette commande, MP-B arrête d'envoyer les données du service au MC-X, et en même temps, enlève les informations de ce client dans sa base de données locale

3) Ensuite, MP-B envoie un message de *KeepAlive* vers tous ses nœuds fils (MC-Y, MC-Z) afin de savoir s'il y a encore un autre client qui utilise le service.

4) Si MP-B n'a reçu aucune réponse de type *Keep-alive Msg Acq*, provenant de ses nœuds fils (MC Y, MC Z), cela signifie qu'aucun client n'utilise le service. Ainsi MP-B arrête d'envoyer les données du service et il enlève les informations des clients Y, Z de sa base de données.

5) MP-B envoie un message de type *LeavingRequest* à son nœud père (MP-A).

6) Un tel processus se prolonge itérativement jusqu'à la racine de l'arbre du réseau SSON, le MS.

Les tasks relatives:

Dans les réseaux overlay existants, tels que XBone, Mbone, RON etc, l'opérateur du réseau est responsable de découvrir les ressources, d'implémenter les composants de service, et de mettre en œuvre la communication entre eux, afin d'établir un réseau overlay. Il contrôle également le cycle de vie du réseau overlay, en termes de processus de création, de maintenance, et de destruction de ce réseau. MBone est un des exemples typiques, dans lequel sa configuration exige une intervention manuelle, et présente plusieurs inconvénients qui ne sont pas acceptables par l'opérateur du réseau ambiant. Bien que quelques outils aient été fourni pour rendre la création de réseau overlay facile, comme XBone, en utilisant l'interface graphique pour manipuler la topologie, et sélectionner les nœuds participants, la durée de cette création est longue et le coût de la gestion est cher. De plus, lors qu'il s'agit d'un réseau de grande échelle, qui comprend de multiples domaines administratifs, concernant différents types de techniques, cette méthode devient plus compliquée.

Par rapport à la gestion de la vie du réseau overlay de façon statique, les réseaux "Skype" "Napster", basés sur des protocoles "peer to peer", proposent une solution dynamique, permettant d'établir un réseau overlay spécifique au service de Voix sur IP ou au service de partage de fichiers de manière distribués dans le réseau Internet. Mais, pour les réseaux mobiles ad-hoc, radio mobile…, il n'y a pas de solution sûre qui va s'adapter aux conditions critiques du réseau, telles que la bande passante (souvent étroite) , l'utilisation des batteries, etc.

Les recherches du SpiderNet[9] s'appuient sur une solution "*hop-by-hop probe processing*", en envoyant la sonde à ses voisions, afin d'obtenir des information sur l'état de ses voisins grâces aux messages de réponse. Cette méthode permet de créer un réseau de service multimédia scalable, autonome et dynamique, et représente une solution semblable au "soft-state" SMART, mais dans ce dernier cas, l'adaptation de service peut être déclenchée par l'utilisateur (client/service fournisseur) ou par un changement du contexte, tel que la dégradation de la QoS du service, la surcharge du réseau, des possibilités du service, la localisation de l'utilisateur…Le "soft-state" SMART présente une plus grande flexibilité et robustesse.

5 Conclusion

Le concept de réseau overlay spécifique au service, présenté dans ce papier, fournit une solution efficace pour fournir un contenu adapté aux utilisateurs, par l'utilisation de nœuds overlays réalisant les traitements applicatifs nécessaires. Ce réseau overlay est capable de s'adapter dynamiquement en fonction du contexte des

utilisateurs et du service, en prenant en compte la mobilité des utilisateurs, le changement de réseau d'accès, de type de terminal…Dans ce cas, le réseau SSON adapte le routage des données dans le réseau overlay, adapte le réseau overlay en ajoutant ou retirant des modules de traitements. Appliqué à ce cas de réseau SSON, la solution de gestion de vie des réseaux overlays de type "soft-state", proposée dans cet article, apparaît comme une solution efficace pour établir, maintenir, adapter ou terminer un réseau SSON. Chaque nœud overlay gère ainsi son propre état mais aussi connaît l'état de ses voisins par des messages de signalisation échangés entre les nœuds. Le cas de gestion de vie, basé sur un arbre de diffusion a été présenté pour illustrer le concept défini. D'autres cas de topologie seront étudiés par la suite.

Remerciements

Ce travail est financé par la Communauté européenne dans le cadre du FP6, dans le projet IST Ambiant Networks, IST-2002-507134. Les auteurs remercient particulièrement les participants du sous-projet 5 du projet ambiant networks, qui ont permis les échanges et les discussions sur ce travail.

Références

[1] Ambient Networks Work Package 1, D1-5, " AN Framework Architecture" IST–2002-507134-AN/ WP1-D05. www.ambient-network .org, 2005

[2] OverQoS: An Overlay Based Architecture for Enhancing Internet QoS, Lakshminarayanan Subramanian, Ion Stoica, Hari Balakrishnan, and Randy Katz , Proc. 1st NSDI, San Francisco, CA, March 2004.

[3] Ambient Networks Work Package 5, D5-1, "SMART: Draft Architecture and Multimedia Routing Decision Logic", IST–2002-507134-AN/WP5/D01, www.ambient-network.org, 2005

[4] Ambient Networks Work Package 5, D5-3, "SMART: Final Architecture Design", IST–2002-507134-AN/WP5/D03, www.ambient-network.org, 2005

[5] David Andersen, hari Balakrishnan, Frans Kaashoek, and Robert Morris "Resilient overlay networks", Proc. 18th ACM SOSP, Banff, Canada, October 2001.

[6] Vinay Kumar, "Mbone: Interactive Multimedia on the Internet," 1995, ISBN 156205937

[7] Joseph D.Touch, Yu-Shun Wang, Venkata Pingali, Lars Eggert, Runfang Zhou, Gregory G. Finn "A Global X-Bone for network Experiments", First International Conference on Testbeds and Research Infrastructures for the DEvelopment of NeTworks and COMmunities (TRIDENTCOM'05), Trento, Italy, 2005.

[8] Christoph Reichert, Michael Kleis, Raffaele Giaffreda, "Towards Distributed Context Management in Ambient Network", IST Mobile and wireless communication Summit, June 2005

[9] XiaoHui Gu, Klara Nahrstedt, "Distributed Multimedia Service Composition with Statistical Qos Assurances", IEEE transactions on multimedia. 2005

A Performance Comparison of Energy Consumption for Mobile Ad Hoc Networks Routing Protocols

Mounir FRIKHA
Tunisian Communication's High School (Sup'Com).
Route Raoued Km 3.5, 283 Ariana
m.frikha@supcom.rnu.tn….
Jamila BEN SLIMANE
Tunisian Communication's High School (Sup'Com).
12 Rue Moussa Iben Noussayr Sidi Daoud La Marsa,
2046 La Marsa Tunis.
Jamilabs07@yahoo.fr

ABSTRACT *One of the main challenges facing Mobile Ad-Hoc network (MANET) is reducing the energy consumption since nodes are usually mobile and battery-operated. Among the most fundamental problems in a Mobile Ad-Hoc Network is the design of efficient routing protocols. So, many different protocols have been proposed in the literature, each one based on a variety of characteristics and properties. Some of these protocols have been studied and their performance has been evaluated in detail focusing on aspects like routing overhead, latency, QoS and route length.*
In this paper, we propose to analyse the energy consumption issues of routing protocols. We present a performance comparison of the following routing protocols: DSR, AODV, TORA and DSDV with respect to energy consumption, evaluating how the different approaches and algorithms affect the energy usage in the mobile devices.
RÉSUMÉ *Un des défis principaux faisant face aux réseaux mobiles Ád-Hoc (MANET) est la réduction de la consommation de l'énergie. Parmi les problèmes les plus fondamentaux dans des tels réseaux est la conception des protocoles de routage. En effet, différents protocoles ont été proposés dont chacun est basé sur un ensemble de caractéristiques et de propriétés. Certains de ces protocoles ont été étudiés et leurs performances ont été évaluées en détail en terme délai de latence, QoS et longueur de route.*
Au niveau de cet article, nous proposons d'analyser le comportement en terme consommation d'énergie de certains protocoles de routage. Nous présentons une comparaison entre les protocoles DSR, AODV, TORA et DSDV tout en mettant le doigt sur l'aspect énergétique.

KEYWORDS: *Ad-Hoc Networks, energy consumption, routing protocols.*

MOTS-CLÉS: *Réseaux Ad-Hoc, Consommation de l'énergie, protocoles de routage.*

1. Introduction

The Mobile Ad Hoc Networks (MANETs) are multi-hop wireless networks where all nodes cooperatively maintain the network connectivity. These types of networks are instantly deployable without any base station or fixed infrastructure. Among the most important MANET features for many researchers we find: dynamic topologies, bandwidth-constrained, variable-capacity links, limited physical security and energy-constrained operations. In our work we mainly focus in the energetic aspect of MANETs routing protocols.

This paper is presented in six parts as follows: the first part of this paper gives an overview of some of the popular Ad Hoc networks routing protocols. In the second part, we present the current protocols that are designed for the device energy conservation. The third part presents the details of the simulation tools and environments. The fourth part presents the methodology we followed to perform simulations whose results are described in the fifth part. Finally, the sixth part presents the conclusion to our work.

2. Overview of ad hoc routing protocols

Various dedicated routing protocols have been proposed to the Internet Engineering task Force (IETF) MANET Working Group. Generally, the routing protocols proposed for MANETs are categorized as table-driven, source-initiated on-demand driven, and hybrid. Some of these protocols have been studied and their performances have been analysed in details.

2.1. *Table-Driven Routing Protocols*

Similar to wired networks, each node in the table-driven routing protocols maintains a routing table containing a list of all the destinations, next hop, and the number of hops to each destination (Morteza and al, 2002). The routing table is constructed using either link-state or distance vector algorithms. Both algorithms require a periodic update of data that must be diffused by different network's nodes.

The advantage of these protocols is that a path to the destination is immediately available, so no delay is experienced when an application needs to send packets. In some cases this can be useful, as for interactive applications.

The main table-driven routing protocols are the following:

The *Optimized Link State Routing* (OLSR) (Said, 2003) protocol is an optimization over the classical link state protocol, tailored for mobile Ad-Hoc networks. The key idea of OLSR is the use of *multipoint relay* (MPR) nodes to flood the network in an efficient way by reducing duplicate packets in the same region. The protocol also selects bidirectional links for the purpose of routing, so that the problem of packet transfer over unidirectional links is avoided.

Destination-Sequenced Distance Vector (DSDV) (Opplige and al, 1998), *Wireless Routing Protocol* (WRP) (Stallings, 1999), and *Global State Routing* (GSR) (Brickell and al, 1996) use destination sequence numbers to keep routes loop free and up to date. These sequence numbers are assigned by the destination node and allow the mobile nodes to distinguish invalid routes from new ones. GSR is similar to the DSDV scheme but uses the link state instead of the distance vector. Each node maintains a link state table based on the information exchanged periodically with the neighbours. The update is selected based on the timestamp of the sequence numbers.

The *Fisheye State Protocol* (FSR) (Yi and al, 2001) is a proactive protocol based on the so-called "fisheye technique," proposed to reduce the size of information required to represent graphical data. The novelties in FSR are the transmission of link-state packets to neighbours instead of flooding method (a method borrowed from the Global State Routing protocol, GSR) and the introduction of scope concept to define network regions with different accuracy in routing information.

Contrary to DSDV and GSR, the protocols *Cluster Gateway Switching Routing* (CGSR) (Buttyan and al, 2000) and *Hierarchical State Routing* (HSR) (Dahill and al, 2001) use hierarchical routing schemes. The CGSR protocol extends DSDV by grouping nodes into clusters. The HSR protocol extends CGSR by forming a clusterhead hierarchy.

2.2. Source-Initiated On-Demand Driven Protocols

This approach is characterized by the elimination of the conventional routing tables at the nodes and consequently the need of their updates to track changes in the network topology.

The main on-demand routing protocols are the following:

The *Cluster Based Routing Protocol* (CBRP) (Marti and al, 2000) is an extension of CGSR where nodes are divided into clusters. When a source has data to send, it floods route request packets only to the neighbouring clusterheads. Upon receiving the request, a clusterhead checks to see if the destination is in its cluster. If so, the request is sent directly to the destination; otherwise, the request is sent to all its adjacent clusterheads.

The *Dynamic Source Routing protocol* (DSR) (Oppliger, 1998) is a typical example of the on-demand protocols, where each data packet carries in its header the complete ordered list of nodes the packet passes through. And this, by having each node maintain a *route cache* that learns and caches routes to destinations. The DSR protocol is composed of two main mechanisms, Route Discovery and Route Maintenance.

The *Ad Hoc On Demand Distance Vector routing* (AODV) (Zhou and al,1999 Perkins and al, 2002) borrows the use of the sequence number from DSDV to supersede stale cached routes and to prevent loops, while the discovery procedure is derived from the one adopted in DSR. Its main difference from DSR is that a

discovered route is locally stored at the nodes rather than included in the packet's header.

The *Temporally Ordered Routing* (TORA) (Park and al, 1997) is a highly adaptive protocol that provides multiple routes for any desired source–destination pair and localizes the control messages to a very small set of nodes near the location of a topological change. To accomplish this, the nodes maintain routing information on adjacent (one-hop) nodes and use a "height" metric to establish a *directed acyclic graph* (DAG) rooted at the destination. When the DAG route is broken during node mobility, route maintenance is necessary to reestablish a DAG rooted at the same destination.

The *Associativity Based Routing protocol* (ABR) (Perkins and al, 1999) is an on-demand protocol carefully designed to work in mobile environments. The key idea is the use of route longevity, instead of the route length, as the main selection criterion. In other words, a high degree of associativity may indicate a low state of node mobility, while a low degree may indicate a high state of node mobility.

2.3. Hybrid Routing Protocols

The hybrid approach combines the table-driven and source initiated on-demand driven approaches such that the overhead incurred in route discovery and maintenance is minimized while the efficiency is maximized. The main hybrid routing protocol is the following:

The *Zone Routing Protocol* (ZRP) (Haas and al, 1998) partitions the network implicitly into zones, where a node zone includes all nearby nodes within the zone radius defined in hops. It applies proactive strategy inside the zone and reactive strategy outside the local zone.

3. Power Conservation Techniques at various Protocol layers

Ad-hoc wireless networks are power constrained since nodes operate with limited battery energy. To maximize the lifetime of these networks (defined by the condition that a fixed percentage of the nodes in the network "die out" due to lack of energy), network-related transactions through each mobile node must be controlled such that the power dissipation rates of all the nodes are nearly the same.

3.1. Physical Layer

In this layer, the energy consumption is due to physical resources: CPU, memories of posting…Researches on the energy consumption on physical layer were mainly focused in the material used. However, the needed energy for the packet radio transmission represents actually the highest energy quantity that a node can spend.

In order to solve such a problem, several solutions were born (Sidi-Mohammed Senouci, 2003). Among these solutions, we find the implementation of a power control policy : the VPT protocol implementation (Variable Power Transmission).

3.2. Data-Link Layer

Collision and congestion control protocols do increase the energy consumption in this layer. In Data link layer, energy can be conserved by adopting the efficient retransmission schemes and making the node turn off (sleep state) when not receiving or transmitting packet.
There are several MAC protocols, but the protocol that does guarantee a significant reduction of the energy consumption is the *Power-Aware Multiple Access protocol with Signalling* (PAMAS) (Singh and al, 2000) protocol. The PAMAS protocol saves from 40 to70% of battery power by intelligently turning off inactive radios.

3.3. Network Layer

In this layer, the energy consumption is essentially due to routing protocols. The greatest class of energy saving protocols belongs to the network layer protocols (Lee, 2001). On network layer, the energy consumption can be limited by decreasing the necessary load relaying the routes. In order to reach such aims, new metrics are proposed. These metrics must:

- Minimize consumed energy /packet,
- Maximize time to network partition,
- Minimize variance in node power levels,
- Minimize cost/packet,
- Minimize maximum node cost.

The fundamental idea allowing a power consumption optimization by the routing process is to route packets according to the minimization of one or several criteria relating to the battery consumption. These criteria are the following:

1. A global minimization (Sidi-Mohammed Senouci, 2003).
2. A local minimization (Ghandour, 2004).
3. A modulation of the emission's power (Sidi-Mohammed Senouci, 2003).

In order to select the best minimum energy routing path having longer overall battery life in the network. There are four different power related schemes (Lee, 2001).

a) Minimum total transmission power routing (MTPR) (Dileep, 2001),
b) Minimum Battery Cost routing (MBCR) (Dongkyun and al,2002),
c) Minimum Maximum cost routing (MMBCR) (Toh, 2001),
d) Conditional Max-Min battery capacity routing (CMMBCR) (Dileep, 2001).

3.4. Transport Layer

In this layer, the communication is featured as end to end. Then, the Ad -Hoc networks can communicate together after forming a route by RREQ technique. In this layer if we avoid repeating retransmission, handling packet in localization manner, and using some power effect routing scheme there is chance for over hear by the other nodes consuming power, these can be avoided by keeping the other nodes, which are not participating actively in routing, in sleep mode.

Among the suggested protocols for such situations, we find the transport protocols Reno and New Reno (Sidi-Mohammed Senouci, 2003).

3.5. Application Layer

Battery energy is exhausted by applications, mainly during data compression and encryption. The energy optimization on the application layer represents a significant field of research. Among the suggested solutions, we quote the principle of data encoding.

4. Simulation Environment

4.1. Energy Consumption Model

According to the specification of the NIC modelled, the energy consumption varies from 230mA in receiving mode to 330mA in transmitting mode, using a 3.3V or 5.0V energy supply (Dongkyun and al, 2002, Carlos Cano and al, 2001). In this work, we have assumed an energy supply of 5V.

4.2. Methodology

The overall aim of this work is to measure and compare the energy consumption behaviour of the four analysed routing protocols. Our basic methodology consists of selecting the most representative parameters for MANET, then defining and simulating a basic scenario and finally simulating and evaluating more scenarios, by varying the selected parameters,.

In the simulation, nodes move according to a model called “random waypoint”. The two factors featuring the Motion are: the maximum speed and the pause time.

As the basic scenario, we have considered a MANET with 25 mobile nodes randomly spread over a500mx500m area. Nodes were moving with a maximum speed of 15 meters/sec with a pause time of 0 seconds. A total of 20 traffic sources have generated CBR data traffic with a sending rate of 4 packets/sec, using a packet size of 512 bytes. Each simulation has a 600 simulated second duration.

5. Simulation results

In this section, we propose to compare the energy consumption for the four routing algorithms: first, over the basic scenario and then over a wide variety of scenarios and traffic models by varying one of the five selected parameters.

5.1. The basic scenario

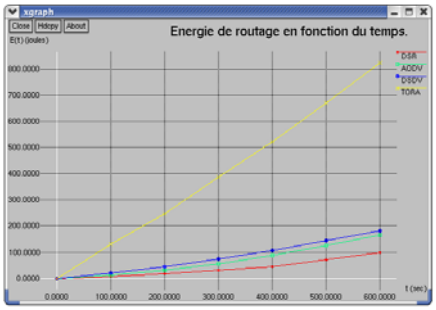


Figure 1. *Routing energy consumption comparison as a function of time.*

As shown in Figure 1, the DSR performs better than AODV and DSDV in terms of energy consumption, and this because of routing protocol packets. DSR performs better than AODV although DSR uses source routing, and AODV hop-by-hop routing (having DSR longer header packets). It is probably due to promiscuous overhearing and caching mechanisms used in DSR to reduce the discovery routes overhead. TORA high-energy consumption is mainly due to the aggregation of IMEP discovery routes packets and TORA maintenance packets.

5.2. Varying motion pattern

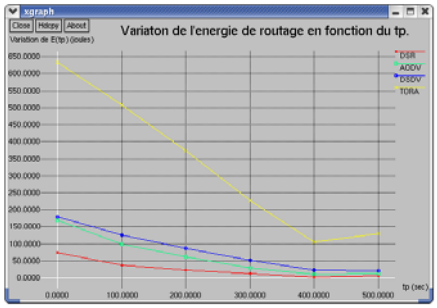


Figure 2. *Routing energy consumption variation as a function of pause time.*

In this section, we explore the effect of varying motion patterns over the basic scenario. We run simulations, varying the pause times from 0, 100, 200, 300, 400 and 600 simulated seconds obtaining a range of scenarios spanning from continuous motion nodes (0 pause time) to static ones (600 pause time). Figure 2 highlights the energy consumed by routing protocols. DSR offers the best performance while TORA shows the worst results. Typically on-demand protocols present an energy descendent trend as the motion rate drops.
Inside the TORA presents the worst index, basically because of the IMEP and TORA packets aggregation.

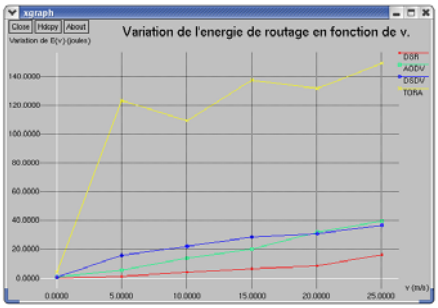


Figure 3. *Routing energy consumption variation as a function of maximum node speed.*

Figure 3 shows the results by varying the maximum node speed among the values 0, 1, 5, 15 and 25 m/s. These values have been selected to simulate the following scenarios : (a) A static network, (b) A MANET of humans walking, (c) A cyclists MANET community, (d) An urban cars MANET and (e) A road cars MANET.
The energy consumption of the on-demand protocols increases as the maximum motion speed grows. As the motion speed changes from a static network scenario to an urban cars MANET, the difference between DSR and AODV grows from a factor of 2.14, to a factor of 3.63. Finally, when the speed grows, DSDV performs better than AODV.

5.3.Varying traffic pattern

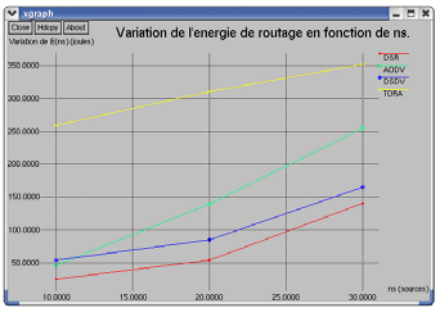


Figure 4. *Routing energy consumption variation comparison as a function of traffic sources.*

Figure 4 shows the behaviour of the four protocols while varying the number of sources between 10, 20 and 30. For on demand protocols such as DSR, AODV and TORA, although the increase of the sources number generates an increment of routing packets, the consumed energy follows a slower shape compared to the sources increment. When traffic sources vary from 10 to 20, the routing energy increases by 114.1% in DSR, 202,18% in AODV and 20,18% in TORA. While moving from 20 to 30 sources the routing energy increases by 160,8% in DSR, 82,3% in AODV and 13,19% in TORA. This is mainly because the on-demand routing protocols allow nodes to learn new route information from packets previously sent.

5.4. Varying node number

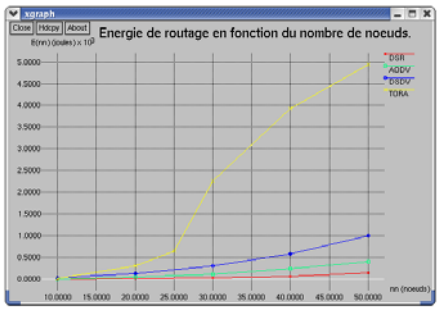


Figure 5. *Routing energy consumption comparison as a function of node number.*

Figure 5 shows the simulation results by varying the number of nodes and maintaining the traffic load. We have selected MANET communities of 10, 20, 25 (basic scenario), 30, 40 and 50 nodes. The TORA behaviour highly depends of this

factor. While moving from 25 to 50 nodes, the protocol suffers an increment of 833%. This characteristic makes this protocol not scalable. With AODV, the energy increment from a MANET of 25 nodes to a MANET of 50 nodes is about 400%. This increment is mainly due to route maintenance process; with DSDV, the increment is mainly due to the propagation of route table between nodes.

5.5. Varying area shape

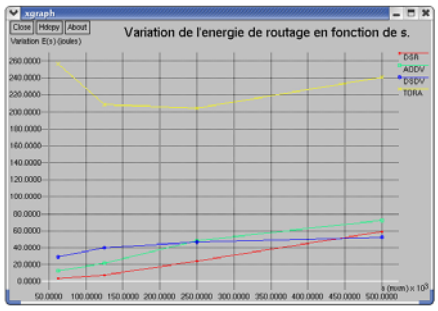


Figure 6. *Routing energy consumption comparison as a function of the MANET area size.*

Figure 6 shows the results while varying the area shape. The selected areas are: 250mx250m, 250mx500m, 500mx500m and 1000mx500m. By incrementing the area, the DSR and the AODV protocols increase their routing energy consumption faster than table driven protocols such as DSDV. In the case that MANET area shape allows long routes (e.g., 500mx500m) table-driven DSDV performs better than on-demand AODV. For a 1000mx500m area scenario, TORA presents again the worst results with respect to the other routing protocols.

5.6. Evaluating data sending pattern

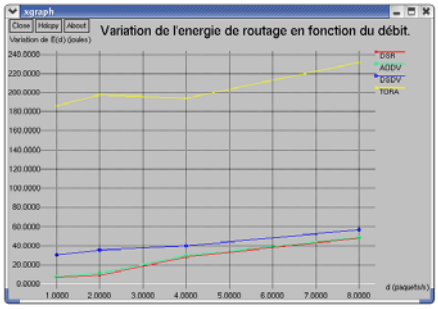


Figure 7. *Routing energy consumption comparison as a function of the MANET sources sending rate.*

Figure 7 shows the effect of increasing the sending rate. This effect is analogous to the effect of increasing the source number. All the protocols quite present a steady behaviour, the DSDV protocol thanks to its table driven scheme, while on demand protocols thanks to their property of learning route information from previous packets.

6. Conclusion

In this work, we presented the results of measuring and comparing the energy consumption behaviour of four routing protocols; respectively the AODV, DSR, the TORA and the DSDV. We selected the most representative parameters for a MANET, then we defined and simulated a basic scenario and finally, by varying the selected parameters, generated and simulated more scenarios.

The simulation results allowed us to conclude the following as far as energy consumption refers. Generally pure on-demand protocols such as DSR and AODV perform better than DSDV, and clearly better than TORA. For all the explored scenarios, TORA has the worst performance index. Besides, increasing the number of nodes while maintaining the number of traffic sources makes TORA not scalable. For example, when nodes move from 25 to 50, the energy consumption increases by 833%. Almost DSDV offers a constant behaviour for all tested scenarios, mainly due to its table-driven philosophy.

The DSR normally performs better than AODV except in static networks in which both show a similar behaviour.

7. References

Brickell E., Feigenbaum J., and Maher D., DIMACS "Workshop on Trust Management in Networks", South Plainfield, NJ, September. 1996.

Buttyan L., and Hubaux J.-P., " Enforcing Service Availability in Mobile Ad-Hoc WANs, Proceedings of the First IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHoc) ", Boston, MA, August. 2000.

Carlos Cano J., and Manzoni P., "A Performance Comparison of Energy Consumption for Mobile Ad Hoc Networks Routing Protocols" pp 1-8, 2001.

Dahill B., Levine B.N., Royer E., and Shields C., " A Secure Routing Protocol for Ad Hoc Networks ", Technical Report UM-CS-2001–037, University of Massachusetts, Amherst, August. 2001.

Dileep K.,"Energy Management in Adhoc Networks". Department of Computer Science, Vaasa University-Filand, pp 1-8, 2001.

Dongkyun K., Garcia-Luna-Aceves J.J., and Obraczka K., "Power-Aware Routing Based on The Energy Drain Rate for Mobile Ad Hoc Network", University of California at Santa Cruz,2002.

Ghandour F., "Optimisation de l'énergie dans les réseaux ad hoc", Report of studies end project, Sup'Com, pp 35-37, June, 2003/2004.

Haas Z., and Pearlman M., "Performance of Query Control Schemes for the Zone Routing Protocol", ACM SIGCOMM, August. 1998.

Lee.M, "Energy-Efficient Routing Protocols in Wireless Ad hoc Networks", Research Project Report, Department City College of City University of New York, 2001.

Marti S., Giuli T.J., Lai K., and Baker M., "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks", Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking, Boston, May, 2000.

Morteza M., Karthik D., and Massoud P., "Power-aware Source routing Protocol for Mobile Ad Hoc Networks," ISLPED'02, Monterey, California,USA, August 12-14, 2002.

Oppliger R., "Internet and Intranet Security", Artech House Publishers, Norwood, May, 1998.

Park.Q, Aslam.J., and De Couto.D., "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks", the Conference on Computer Communications (IEEE infocom), Kobe, Japan, April 1997.

Perkins C. E., Royer E. M., and Das S. R., "Ad Hoc on Demand Distance Vector (AODV) Routing". IETF Internet Draft, Juin 2002. (work in progress).

Perkins C., and Royer E., "Ad-hoc On-Demand Distance Vector Routing", 2nd IEEE Workshop on Mobile Computing Systems and Applications, February. 1999.

Said N., "Etude comparative des protocoles de routage dans les réseaux Ad hoc", Report of studies end project, Sup'Com, 2002/2003.

Sidi-Mohammed Senouci M., "Applications de Techniques d'Apprentissage dans les Réseaux Mobiles", Doctorate thesis, Université de Pierre et Marie Curie, pp 108-109, October 2003.

Singh S., and Raghvendra C.S.. " PAMAS – Power Aware Multi-Access protocol for Signalling for Ad-Hoc Networks",ACM Computer Communications Review, 1999- 2000.

Stallings W., "Cryptography and Network Security", 2nd Ed., Prentice Hall, Englewood Cliffs, NJ, 1999.

Toh C.K., " Maximum Battery Life Routing to Support Ubiquitous Mobile Computing in wireless Ad hoc Networks ", IEEE Communication Magazine, June 2001.

Yi S., Naldurg P., and Kravets R., "Security-Aware Ad Hoc Routing for Wireless Networks", Technical Report UIUCDCS-R-2001–2241, August. 2001.

Zhou L., and Haas Z.J., "Securing Ad Hoc Networks", IEEE Network Magazine, Nov./Dec. 1999.

Vers une solution au problème d'exclusion mutuelle dans les réseaux mobiles ad hoc

M. Benchaïba* ¾ **M. Haddad **** ¾ **M. Ahmed-Nacer*,**

**LSI-Département Informatique- USTHB- BP. N°32, El-Alia, Bab Ezzouar, Alger – Algérie,*

benchaiba@lsi-usthb.dz
anacer@mail.cerist.dz

*** Laboratoire PRISMa - Université Claude Bernard Lyon 1, Bât. Nautibus (ex 710), 843, Bd. du 11 nov. 1918, 69622 Villeurbanne Cedex – France,*

mhaddad@bat710.univ-lyon1.fr

RÉSUMÉ: *Cet article propose un nouveau protocole d'exclusion mutuelle pour les réseaux mobiles ad hoc utilisant une approche à jeton. Ce protocole ne repose pas sur la couche de routage et, pour la circulation du jeton, privilégie la satisfaction des demandes en même temps selon leurs anciennetés et leurs distances par rapport au jeton. Une requête d'un processus est diffusée selon un rayon dynamique permettant d'atteindre une proportion de nœuds dont une requête est présente localement. Par ailleurs, le protocole proposé prend en charge quelques caractéristiques des réseaux mobiles ad hoc telles que : le mouvement des nœuds (i.e. formations et pertes de liens), les arrivées de nouveaux nœuds, leurs départs et leurs pannes. Il tolère aussi des partitionnements temporaires du réseau mais ne permet pas la perte du jeton.*

ABSTRACT: *In this paper, we describe a new token based mutual exclusion protocol for mobile ad hoc networks. This protocol does neither use the routing layer nor a logical structure and agrees requests based on their distances away to the token and their olds. A request is broadcasted with a dynamical radius calculated so to reach a part of the nodes for which a request is present in the local queue. The protocol traits some characteristics of mobile ad hoc networks: node movements (i.e. creates and fails of links), connection, disconnection and fail of nodes. Also, the protocol supports temporary partitioning of the network but does not support the token loss.*

MOTS_CLÉS : *réseau mobile ad hoc, exclusion mutuelle, section critique, jeton.*

KEYWORDS: *mobile ad hoc network, mutual exclusion, critical section, token.*

1. Introduction

L'exclusion mutuelle (EM) est un problème fondamental dans les systèmes distribués. Ce problème est rencontré lorsqu'on est en présence d'un ensemble de processus qui demandent, de façon simultanée, l'accès à une même ressource à travers une parcelle de code appelée section critique (SC). Au plus, un processus peut entrer en SC à un moment donné. Toute solution au problème d' EM doit garantir deux conditions essentielles : la *sûreté* impliquant l'existence d'un processus au plus au niveau de sa SC ; la *vivacité*, impliquant que pour toute exécution infinie du protocole, chaque processus pourra accéder à sa SC autant de fois qu'il le souhaite. Le problème d' EM a été largement traité dans les réseaux statiques distribués. Ainsi, plusieurs solutions ont été proposées (Singhal, 1993), selon deux approches essentielles : l'approche à *permission* et l'approche à *jeton*, dont quelques-unes ont été adaptées pour fonctionner dans les réseaux cellulaires ainsi que dans les réseaux mobiles ad hoc.

Dans l'approche à jeton, deux méthodes sont couramment utilisées et sont liées à la *demande du jeton* et à la *circulation de jeton*. Dans la première méthode, un nœud demandant la SC est amené à avoir le jeton ; le problème de base est comment l'atteindre. Dans certains algorithmes, la requête est adressée à tous les nœuds du réseau (Ricart *et al*, 1983); dans d'autres, une structure logique est définie pour désigner le détenteur du jeton, par exemple un DAG (Direct Acyclic Graph) (Raymond, 1989): la requête est envoyée le long d'une branche du DAG en direction du nœud détenteur du jeton. Dans (Ricart *et al*, 1983), Ricart et Agrawala proposent un algorithme, sur une topologie complète de réseau, qui requiert au maximum N messages pour réaliser l' EM. Suzuki et Kazami (Suzuki *et al*, 1985) proposent un algorithme, basé sur celui de Ricart et Agrawala, dans lequel la file de requêtes est transportée par le jeton. Singhal (Singhal, 1989) a amélioré les performances de l'algorithme de Suzuki et Kazami jusqu'au maximum N messages dans le cas des charges élevées en utilisant une heuristique pour déterminer l'ensemble de nœuds qui peuvent posséder le jeton. Dans (Raymond, 1989), Raymond a développé un algorithme, basé sur une structure d'arbre logique dont la racine est le nœud détenteur du jeton, qui requière au maximum $O(Log\ N)$ messages pour entrer à la SC. Dans (Chang *et al*, 1990), Chang *et al* ont développé un algorithme qui remédie aux insuffisances de celui de Raymond notamment par la tolérance aux pannes de liens et de nœuds en maintenant des chemins multiples pour atteindre le jeton. Dans (Dhamdhene *et al*, 1994), Dhamdhene et Kulkarni ont développé un algorithme dont le but est de résoudre le problème du cycle resté encore non résolu dans l'algorithme de Chang *et al* et il est *k-résistant*, c'est à dire tolère k pannes de nœuds/liens. Dans la méthode de circulation de jeton, notons seulement la solution proposée par Le Lann (Le Lann, 1977). Dans cette solution tous les nœuds sont logiquement organisés en un *anneau unidirectionnel* et le jeton *circule* selon cet anneau.

Dans l'approche à permission, pour entrer à sa SC, un nœud demandeur doit attendre de recevoir les réponses de tous les autres nœuds (ou d'un sous-ensemble dans certains algorithmes) du réseau. Dans l'algorithme proposé par Lamport (Lamport, 1978), quand un nœud veut entrer à sa SC, il envoi une requête à tous les autres nœuds et attends leurs réponses. A la sortie de sa SC, il envoi un message de libération à tous les autres nœuds. Cet algorithme requière *3(N-1)* messages par entrée en SC. Dans (Ricart *et al*, 1981), Ricart et Agrawala ont proposé un algorithme qui requière *2(N-1)* messages par entrée en SC. Quand un nœud veut entrer à sa SC, il envoi une requête à tous les autres nœuds et attend leurs réponses. A la réception de l'agrément de tous ces nœuds, il entre à sa SC. Dans (Carvalho *et al*, 1983), un processus qui désire entrer en SC, n'envoie un message estampillé qu'aux processus qui ont accédé à la SC depuis sa dernière demande. Le nombre de messages requis est inférieur ou égal à *2(N -1)* Dans (Maekawa, 1985), Maekawa a développé un algorithme qui requière *c(ÖN)* messages pour l'accès à une SC. L'algorithme se base sur les structures de *quorums* permettant d'associer à chaque nœud un ensemble de nœuds desquels seulement la permission d'accès à la SC est demandée.

Les algorithmes initialement prévus pour les réseaux fixes ne peuvent pas s'appliquer aux environnements mobiles (Benchaïba *et al*, 2004), mais en sont en général une adaptation. Dans (Badrinath *et al*, 1994), B. R. Badrinath *et al* ont proposé deux algorithmes distribués d' EM pour les réseaux cellulaires. Le premier est une adaptation pour les réseaux cellulaires de l'algorithme proposé par Lamport (Lamport, 1978), le second est une adaptation de l'algorithme proposé par Le Lann (Le Lann, 1977). Dans (Walter *et al*, 2001), Walter et Kini ont proposé un algorithme distribué d' EM pour les réseaux mobiles ad hoc orienté jeton dérivé de plusieurs autres algorithmes : le protocole de routage de Gafni et Bertsekas (Gafni *et al*, 1981) basé sur un arbre, l'algorithme présenté dans (Chang *et al*, 1990), et d'autres idées de (Dhamdhene *et al*, 1994) et (Raymond, 1989). Cet algorithme définie une structure bâtie sur la topologie réelle du réseau qui est représentée par un DAG de pointeurs orientés jeton, maintenant plusieurs chemins menant vers le détenteur du jeton. Cet algorithme s'adapte bien à l'environnement mobile ad hoc puisque il requière aux nœuds de maintenir uniquement des informations sur les voisins immédiats. Quand un processus est amené à véhiculer le jeton au prochain, il trouve nécessairement un chemin de retour puisque le non partitionnement n'est pas permis. Cela suppose que le chemin vers le détenteur du jeton est toujours entretenu à l'avance. Dans (Baldoni *et al*, 2002), Baldoni et Virgillito ont proposé un algorithme utilisant un anneau logique dynamique et combinant la méthode de recherche et celle de circulation de jeton. L'algorithme essaye de réduire la consommation d'énergie par la réduction du nombre de pas traversés pour l'accès à une SC et évite d'envoyer des messages de contrôle dans le cas d'absence de requêtes. La mobilité est supportée à l'aide de l'utilisation de la couche de routage, cela permet de sélectionner le nœud le plus proche pour l'envoi du jeton. Dans (Malpani *et al*, 2005), Malpani et al ont proposé un algorithme orienté jeton, paramétrable et avec plusieurs variantes. Le jeton circule *continuellement* sur un

anneau logique dynamique de taille variable. Toutes les variantes ont le même squelette mais diffèrent sur la politique de *sélection* du prochain nœud à visiter.

Dans cet article, nous proposons un nouveau protocole d'EM pour les réseaux mobiles ad hoc basé sur l'approche à jeton (qui est d'ailleurs la tendance actuelle). Le protocole décrit dans cet article ne repose pas sur le routage, n'utilise pas de structure logique de communication et qui, pour la circulation du jeton, privilégie la satisfaction des demandes en fonction d'une part de leur ancienneté et d'autre part de leur distance par rapport au jeton. Le protocole favorise la scalabilité du réseau et considère un nombre indéterminé de nœuds. Par ailleurs, le protocole proposé prend en charge quelques caractéristiques des réseaux mobiles ad hoc telles que : le mouvement des nœuds (i.e. les cassures et formations de liens), les arrivées de nouveaux nœuds, leurs départs et leurs pannes. Il tolère aussi des partitionnements temporaires du réseau mais ne permet pas la perte du jeton.

Cet article est organisé comme suit : Après un bref aperçu sur quelques travaux antérieurs, dans la section 2 nous présentons les idées de base du protocole, les structures de données et les messages utilisés. Ensuite dans la section 3, une description du protocole est donnée à travers les différents évènements qui le constituent. Un simple exemple d'exécution suivi d'une discussion sur certains aspects cachés du protocole sont donnés dans la section 4. Et finalement, la section 5 conclue l'article. Le texte du protocole est décrit à l'annexe.

2. Préliminaires

2. 1. Modèle du réseau

Nous supposons un réseau composé d'un nombre *variable* de nœuds mobiles, chaque nœud possède un identificateur unique. Les nœuds utilisent un support de communication sans fil, les liens de communication sont bidirectionnels, FIFO et fiables. Un nœud peut tomber en panne, quitter ou intégrer le réseau, à tout moment. Les mouvements des nœuds peuvent engendrer des créations, des pertes de liens et arrivées, départs et pannes de nœuds. Cependant, seul le partitionnement bref est toléré. La couche de liaison permet aux nœuds de connaître leurs voisins à tout moment. Dans la suite de l'article et afin de décrire notre protocole, nous nous plaçons au niveau du nœud (ou processus) désigné par l'indice ***i***.

2. 2. Idées de base du protocole

Le protocole utilise l'approche à jeton dont l'unicité du jeton dans le réseau assure d'une manière intrinsèque l' EM dans celui-ci. Il combine la circulation et la demande de ce dernier. Le jeton transporte entre autres une file de requêtes récoltées

lors de son parcourt. Le protocole se base sur la connaissance locale, celle du voisinage immédiat et ne prend en considération que les liens physiques pour acheminer de l'information. Ce qui lui permet de favoriser la *scalabilité* du réseau et de considérer un nombre *indéterminé* de nœuds. D'autre part, le protocole fait recours à un certain nombre de mécanismes qui sont :

2.2.1. Rayon de diffusion

Puisque la détention du jeton conditionne l'accès à la SC, la diffusion d'une requête est un des moyens permettant de l'acquérir. Afin de minimiser l'impact de la diffusion, nous introduisons le concept du rayon de diffusion. Ce rayon est calculé de manière dynamique à partir des requêtes déjà reçues par ***i***. Plus précisément, seuls la date de réception de la requête et le nombre de sauts séparant son émetteur de ***i*** servent à calculer ce rayon. Les nœuds se trouvant dans la zone délimitée par ce rayon vont recevoir cette demande. Dans la figure 1(a), admettons que le nœud ***1*** désire émettre une requête, il doit donc calculer son rayon de diffusion R_1. Pour se faire, supposons que sa file Q_1 contient les demandes *valides* (voir plus loin,) des nœuds ***2, 4, 11*** et ***12*** avec leurs routes associées. Avec $R_1 = 3$, il atteindra tous ces nœuds ou avec $R_1 = 2$, il attendra ***50 %*** de ces nœuds. Cette méthode de calcul du rayon favorise la création d'une cascade de chemins vers le détenteur du jeton. Puisque le jeton récolte les requêtes rencontrées sur son chemin, tous ces processus seront éventuellement satisfaits, en regardant la mobilité et les pannes de nœuds.

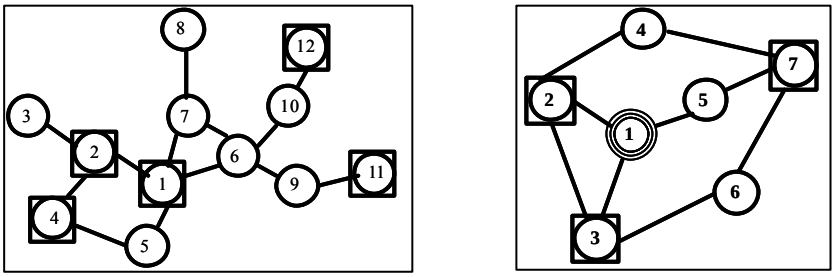


Figure 1. (a) Rayon de diffusion **(b)** Situation de famine

2.2.2. Sélection du nœud privilégié

Chaque nœud recevant une demande aura connaissance de l'identité de son émetteur, ainsi que la route parcourue et l'inscrit dans sa file d'attente. Cette file peut être organisée suivant une politique qui privilégie les requêtes les plus proches. Cette politique induit des cas de famine. En effet, dans le cas d'un groupe de nœuds proches les uns des autres qui demandent l'entrée en SC fréquemment, le jeton ne cessera pas de circuler entre eux, ce qui prive les demandeurs les plus éloignés. Dans la figure 1(b), admettons que les nœuds *2*, *3* et *7* sont demandeurs de SC, que le

nœud *1* est en SC et les nœuds *1*, *2* et *3* demandent continuellement le jeton. En considérant uniquement le nombre de sauts, le jeton sera monopolisé par les nœuds *1*, *2* et *3* et ainsi le nœud *7* sera privé de sa SC. Afin d'éviter ce problème, l'ancienneté des requêtes est aussi considérée (Lamport, 1978). D'où, la priorité d'une requête d'un nœud *j*, au niveau du nœud privilégié ***i*** est donnée par $\boldsymbol{Pr_j = (NS_j + nb\_hops, j),}$ où ***nb_hops*** est le nombre de sauts séparant ***i*** de ***j***. Comme la distance qui sépare deux nœuds est bornée, et que les estampilles des nœuds servis augmenteront rapidement, la priorité tournera donc en faveur des nœuds plus éloignés.

2.2.3. Verrouillage du jeton

Quand un nœud est sélectionné comme prochain à être servi, le jeton lui est véhiculé sur le chemin enregistré avec la requête. Afin d'éviter au jeton de faire éventuellement plusieurs sauts avant d'atteindre son destinataire, et par conséquence améliorer le temps d'attente d'un nœud demandeur, des nœuds intermédiaires demandeurs de SC et se trouvant sur la route du jeton vers le destinataire peuvent être servis, puisque le jeton leur est parvenu mais sans influencer la trajectoire initiale du jeton. Néanmoins, la mobilité des nœuds peut causer des cassures de route ou leur rallonge, le fait que le jeton soit ralenti dans sa route peut créer des cas de famine. Pour remédier à ce problème, la notion de verrouillage du jeton est introduite. Puisque un nœud demandeur non satisfait au bout d'un temps fixé est amené à relancer sa demande un certain nombre de fois, un seuil dit de verrouillage (***Lock_limit***) est définit. Arrivant à ce seuil, le jeton est verrouillé pour sa destination ce qui implique que les nœuds intermédiaires ne peuvent pas entrer en SC.

2.2.4. Recherche de route

Pendant la circulation du jeton vers une destination donnée, la route peut être perdue. D'où l'association d'un état au jeton, ***ONRoute*** s'il suit normalement une route prédéfinie, ***NORoute*** dans le cas de perte de route et un nouveau destinataire n'est pas encore trouvé. Un nœud ***i*** qui reçoit le jeton, s'il est demandeur et le jeton lui est destiné ou il est non verrouillé ou avec un état ***NORoute***, entre en SC, sinon son rôle se réduit à router le jeton suivant la route inscrite sur ce dernier. Dans le cas où cette route serait brisée, le nœud ***i*** tente de la recalculer d'une manière locale, ceci en vérifiant la présence des nœuds suivants le saut brisé dans son voisinage. Si cette opération, échoue le jeton change de destinataire qui sera choisi selon la méthode de *sélection du nœud privilégié.* En cas d'échec, il est envoyé à l'un des voisins avec l'état ***NORoute*** pour la recherche d'un destinataire; la conséquence en est éventuellement la non satisfaction de certains nœuds demandeurs, ce problème peut être surmonté par la possibilité de redemande de SC après un délai fixé.

3. 3. Structures

3.3.1. Structures de base locales à un nœud i

NS$_i$: estampille locale générée par le nœud ***i*** - ***Request_NS$_i$*** : estampille de la dernière requête, sert à garder la même date (i.e. même priorité) pour plusieurs tentatives de la même requête afin d'éviter la famine - ***State$_i$*** : (***Idle***, ***Requesting***, ***InCS, Unable***) : état du nœud ***i*** - ***Holder$_i$*** : indique si le nœud ***i*** est détenteur du jeton ou non, initialement pour ***i=0 Holder$_i$*** = ***True***, autrement ***Holder$_i$*** = ***False*** - ***Last_Holder$_i$*** : désigne le dernier porteur du jeton - ***N$_i$*** : ensemble des voisins de ***i*** - ***R$_i$*** : rayon de diffusion de ***i*** - ***Nb_attempts$_i$*** : nombre de tentatives d'une même demande de SC - ***Ignored_Neighbors$_i$*** : ensemble des voisins ignorés par ***i***, lors de la phase du traitement des cassures de routes, afin d'éviter un problème de boucle - ***Q$_i$*** : file d'attente de ***i***, au format ***(j, Route$_j$, NS$_j$, Valid$_j$, Nb_attemps$_j$).*** Ce format représente la demande d'un nœud ***j***, où ***Route$_j$*** est une séquence des nœuds intermédiaires entre ***i*** et ***j***, ***NS$_j$*** est le numéro de séquence de la demande de ***j***, ***Valid$_j$*** représente la validité de la requête de ***j*** (sert uniquement dans le calcul du rayon de diffusion de ***i***) et ***Nb_attemps$_j$*** est le nombre de tentatives associées à cette requête.

2.3.3. Les messages

– Le message de demande d'entrer en SC : ***Request (i, NS$_i$, R$_i$, Route$_i$, Nb_attempts$_i$, aware$_i$)***, où ***NS$_i$*** est l'estampille de la demande ; ***R$_i$*** est le rayon de propagation selon ***i*** associé à cette demande ; ***Route$_i$*** est la route qui mène à ***i*** (elle sera construite au fur et à mesure que la demande de ***i*** passe par chaque nœud intermédiaire); ***Nb_attempts$_i$*** est le nombre de tentatives d'accès en SC qu'a effectué le nœud ***i*** avec le même numéro de séquence (NS) ; ***aware$_i$*** est l'ensemble des nœuds qui sont au courant de la demande de ***i*** le long de la branche de parcours. Cet attribut accompagne tous les messages destinés à la diffusion afin de minimiser le nombre de messages.

– Le message jeton : ***Token (Token_State, Token_Route, Token_Req_Set, Token_NS, Token_Last_Holder, Locked),*** où ***Token_State*** est l'état du jeton, ***ONRoute*** si le jeton suit un chemin déterminé vers une destination donnée, ***NORoute*** dans le cas de perte de route ; ***Token_route*** est l'ensemble des nœuds constituant la route que doit suivre le jeton pour atteindre une destination donnée ;***Token_Req_Set*** est une vue partielle de l'ensemble des demandeurs de SC dans la partition ; ***Token_NS*** est l'ensemble des numéros de séquences de nœuds ;***Token_Last_Holder*** est le dernier nœud porteur du jeton ; ***Locked*** indique si le jeton est verrouillé ou non pour un nœud destinataire.

– Le message de contrôle : ***LinkInfo(i, Request_NS$_i$, Nb_attempts$_i$),*** émis par le nœud ***i*** après la création d'un nouveau lien seulement s'il est demandeur de SC. Les arguments sont ceux de la requête de ***i***.

3. Description du protocole

Le texte du protocole est formé d'un ensemble de primitives liés aux différents événements le constituant. Les primitives et les routines du protocole sont décrites à l'annexe. Au niveau de chaque nœud, les primitives s'exécutent de manière atomique, cependant des événements peuvent se produire durant l'exécution de la SC.

Quand un nœud ***i*** désire accéder à sa SC, il y accède directement dans le cas de détention du jeton. Autrement, il diffuse une requête ***Request(..),*** estampillée selon les horloges de Lamport (Lamport, 1978), dans son voisinage selon un rayon calculé en fonction des requêtes déjà reçues et dont les routes sont *valides* (voir l'exemple de la figure 1). La route d'une requête est valide pendant un délai ***ValidTimer*** à partir du moment de sa réception, en regardant la mobilité des nœuds du réseau. Dans le cas où la file de requêtes Q_i est vide ou toutes les routes des requêtes présentes sont invalides, le rayon est considéré à priori comme *infini* et c'est aux récepteurs de la requête d'en définir un selon l'état de leurs files. Un délai de garde ***RequestTimer*** est associé à chaque requête émise. Au bout de ce délai, si le jeton ne parvient pas à ce nœud demandeur, la requête est relancée mais avec la même estampille pour lui assurer son ancienneté. Dans le cas échéant, ce processus est exécuté un nombre de fois limité (i.e. ***Request_Threshold***) avant d'observer un échec.

Quand une requête d'un nœud ***i*** est reçue au niveau de ***j***, la fonction ***AddRequest*** l'insère dans la file Q_j si elle n' y figure pas. Dans le cas contraire, seuls ses attributs sont met à jour mais à condition que la requête est plus récente que celle existante ou sa route est meilleure. La structure $Received\_NS_j$ permet d'éviter d'insérer une requête qui ne devrait pas l'être (cette situation peut se produire dans le cas où la requête qui vient d'être reçue a été déjà satisfaite après le dernier passage du jeton au niveau de ***j***). Si le nœud ***j*** possède le jeton, alors il met à jour les informations qu'il transporte pour y inclure éventuellement cette requête. De plus si le jeton était au repos, alors ***j*** le passe au prochain à travers la procédure ***VehicleToken()*** selon la stratégie de *sélection du nœud privilégié*. Dans le cas contraire (i.e. ***j*** ne possède pas le jeton, ou bien il le possède mais il est dans sa SC), il examine la possibilité de faire propager la requête à son voisinage non encore visité par la requête (selon la connaissance reçue avec celle-ci). Dans le cas où $R_j = \infty$ (infini), un rayon local est calculé, de la même manière que lors de la demande d'entrée en SC, suivant le contenu de Q_j.

Quand un nœud ***i*** reçoit le jeton ***Token(..)***, il met à jour ses données en l'occurrence sa file d'attente Q_i en supprimant les requêtes déjà satisfaites et en insérant les nouvelles requêtes inscrites dans le jeton et qui ne lui sont pas parvenues, mais sans aucune information sur les routes correspondantes. Puis, le nœud ***i*** met à jour le jeton en lui intégrant les requêtes non encore inscrites. Ensuite, si le nœud ***i*** est dans un état ***Idle*** il réajuste son horloge dans le jeton pour permettre aux autres nœuds de supprimer les anciennes requêtes propres à ce nœud.

Maintenant, deux cas se présentent : le jeton est dans l'état ***ONRoute*** qui veut dire qu'il circule normalement sur sa route, si le jeton est destiné à ***i*** ou possède l'attribut ***Locked=Faux*** et ***i*** est demandeur alors il entre en SC, sinon il fait passer le jeton au prochain à travers ***VehicleToken()***. L'autre cas est que le jeton est dans l'état ***NORoute***, le nœud ***i*** ajoute l'identité du nœud émetteur du jeton dans l'ensemble $Ignored\_Neighbors_i$ (ce qui permet d'éviter au jeton de circuler en boucle). Dans ce cas, si ***i*** est demandeur il profite de cette situation pour exécuter sa SC, sinon il fait appel à ***VehicleToken()***.

Quand la procédure ***VehicleToken()*** est invoquée, le jeton doit être acheminé vers un prochain destinataire (sauf si le nœud ***i*** est isolé, auquel cas il attend la réception de nouvelles requêtes). Deux cas se présentent, soit le jeton suivait une route déjà prédéfinie par un autre nœud et donc le nœud ***i*** doit poursuivre l'acheminement du jeton sur cette route, soit le nœud ***i*** vient de sortir de la SC, et donc il sélectionne le prochain nœud à satisfaire. Dans le premier cas, le jeton est passé au destinataire s'il est un voisin sinon il est transmis à un voisin proche du destinataire et figurant dans son chemin. Dans le cas d'échec (i.e. perte totale de route), il sélectionne un prochain destinataire, s'il y a lieu à partir de sa file Q_i, selon la technique de *sélection du nœud privilégié*. Pour tous les cas précédents, le jeton est transmis avec l'état ***ONRoute***. En plus, le jeton est verrouillé s'il s'agit d'un nouveau destinataire dont le nombre de tentatives a atteint le seuil ***Lock_Limit***. Si un échec est observé après toutes ces tentatives, le jeton est affecté de l'état ***NORoute*** et il est passé à un voisin qui est sélectionné, afin de rechercher un nouveau destinataire, en évitant les boucles.

A la sortie d'un nœud ***i*** de sa SC, il fait passer le jeton au prochain nœud demandeur (***VehicleToken()***), s'il existe. Dans le cas où la file Q_i est vide alors le jeton est gardé au repos.

4. Exemple d'exécution et discussion

4.1. Exemple d'exécution

La figure2 montre un scénario d'exécution du protocole d'une manière qui permet de faciliter la présentation. Initialement (fig. 2(a)), le nœud *4* est porteur du jeton et les nœuds *2* et *7* veulent entrer en SC. On suppose que leurs estampilles respectives sont égales, mais leurs rayons de diffusion respectifs sont $R_2 = 1$ et $R_7 = 3$ (on note que la requête du nœud *7* atteint le nœud *2*, donc ce dernier possède une route vers le nœud *7*, mais l'autre sens n'est pas vérifié. Le nœud *7* n'ayant aucune connaissance de la demande du nœud *2*, ne possède aucune information sur la route qui mène à ce nœud) (fig. 2(b)). Puisque les deux requêtes arrivent simultanément avec la même estampille, c'est le nombre de sauts qui va déterminer l'ordre d'entrée en SC. Le nœud *2* étant le plus proche du nœud *4*, ce dernier le choisit comme

prochain, et lui passe le jeton (fig. 2(c)). Entre temps, le nœud *1* sort de la portée du nœud *2* et *3* et entre dans la portée du nœud *7* , tandis que le nœud *3* entre dans la portée du nœud 2, mais sans sortir de la portée du nœud *7*. Quand le nœud *2* sort de sa SC, il veut faire passer le jeton au nœud *7*, mais la route initialement construite (i.e. *1*, *3*, *7*) a été brisée, et donc le nœud *2* tente de la recalculer, la nouvelle route (i.e. *3*, *7*) sera obtenue (fig. 2(d)), et le jeton sera véhiculé via cette route vers le nœud *7* (fig. 2(e)).

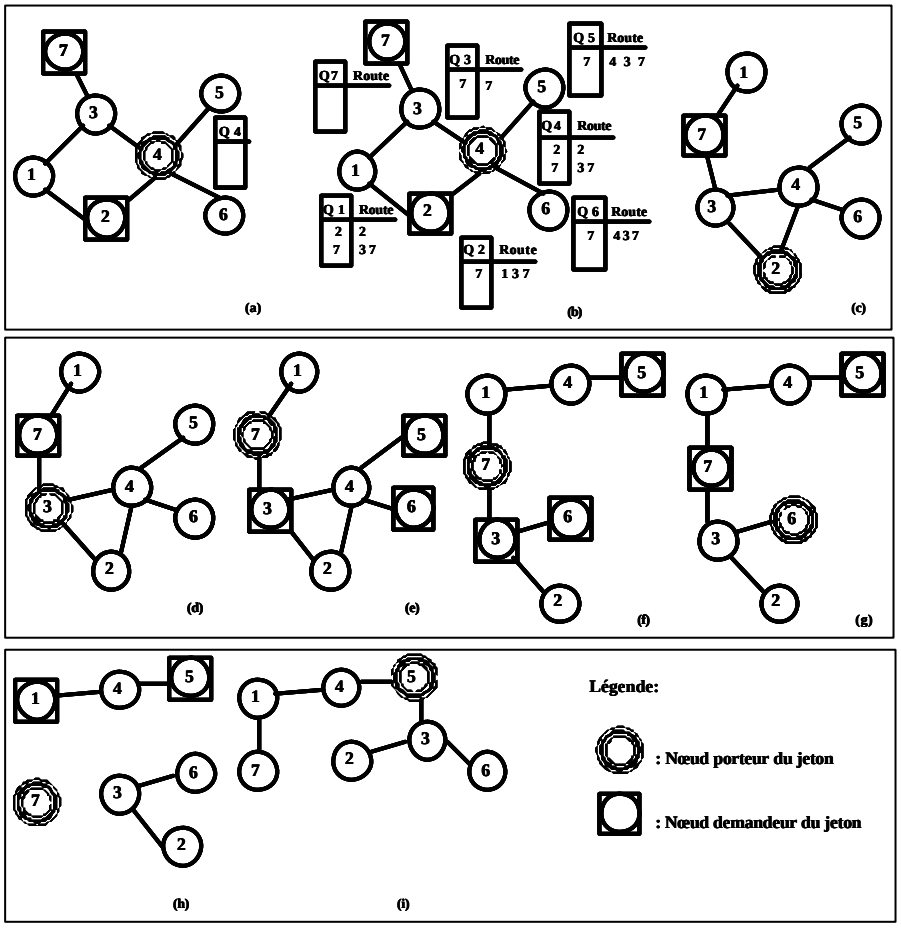


Figure 2. *Un simple exemple d'exécution.*

Dans la figure 2(e), les nœuds *3*, *5* et *6* sont demandeurs de jeton, supposons que $R_3 = 1$, $R_5 = 3$ et $R_6 = 3$. Leurs requêtes arrivent au jeton avec les routes respectives *3* et *3*, *4*, *5* et *3*, *4*, *6*. Dans la figure 2 (f), on suppose que le nœud *6* est le prochain destinataire, le jeton lui est véhiculé via le nœud *3* avec une route recalculée. Arrivant au nœud *3* et puisque il est demandeur (admettons que le nœud *6* n'a pas atteint le seuil de verrouillage), il profite de l'arrivée du jeton et entre à sa SC et à sa sortie il fait passé le jeton à son destinataire déjà prévu (i.e. le nœud *6*) (fig. 2(g)). Dans la figure 2 (g), le nœud 7 demande le jeton avec $R_7 = 1$ par exemple. A sa sortie de la SC, le nœud *6* qui a déjà reçu la requête du nœud 5 avec la route *4*, *5* n'arrive pas à lui passer le jeton. Le jeton est donc envoyé au nœud *3* (i.e. $Last\_Holder_6$) avec l'attribut ***NORoute,*** qui à son tour l'envoi au nœud *7*, mais avec l'attribut ***ONRoute*** puisque il a déjà reçu sa demande. Le nœud 7 utilise le jeton pour exécuter sa SC. Dans la figure 2(h), le nœud *7* est isolé temporairement ; à sa sortie de la SC, il garde donc le jeton à son niveau. Admettons que le nœud *1* exprime une demande avec $R_1 = 2$, sa demande n'atteint pas le jeton. Dans la figure 2 (i) le réseau c'est fusionné en une seule partition. En autres, le nœud *1* détecte un lien avec le nœud 7 et lui envoi sa requête à travers le message ***LinkInfo(..),*** ce qui amène le nœud 7 à lui passer le jeton. En supposant que ***RequestTimer*** au niveau du nœud *5* a expiré et ce nœud a relancé sa demande, le jeton lui sera donc éventuellement véhiculé.

4.2. Discussion

– Quelle est l'utilité de la structure $Received\_NS_i$?

Quand un nœud i sélectionne son prochain pour recevoir le jeton, celui-ci est supprimé de sa file de requêtes. Lors de la réception d'une même requête de ce même nœud, il l'insère dans sa file automatiquement sans savoir si elle est ancienne ; ceci peut engendrer l'envoi non nécessaire du jeton. D'où l'utilisation de la structure $Received\_NS_j$ pour purger les requêtes tardives. Une autre solution pourrait être de transporter dans le jeton les derniers numéros de séquence des requêtes satisfaites de tous les nœuds, mais la taille du message devient considérable.

– Quelle est l'utilité de l'ensemble ***IgnoredNeighbors*** ?

Soit l'exemple de la figure 3(a) : Le destinataire du jeton est le nœud *5* avec le chemin *2 3 4 5*. Arrivant au nœud *3* (figure 3 (b)), le lien avec le nœud *4* est rompu. Le nœud *3* ne peut plus router le jeton vers son destinataire. Supposons aussi qu'il ne trouve pas de nouveau destinataire (i.e. file vide), il décide donc de le redonner à $Last\_Holder_3$ qui est *2*, *2* reçoit le jeton avec ***Token_State*** égal à ***NORoute***, alors il choisit un nouveau destinataire qui est le nœud *6* à partir de sa file Q_2. Il renvoi donc le jeton au nœud *3* (qui est sur la route du nouveau destinataire) avec ***Token_State*** égal à ***ONRoute***, ce qui implique que *2* devient $Last\_Holder_3$. Le nœud *3* n'ayant toujours pas de route valide remet le jeton à *2*, ainsi la boucle est formée et le jeton

bouclera entre *2* et *3*. Donc, dès que *3* rend le jeton à *2*, ce dernier met *3* dans l'ensemble ***IgnoredNeighbors***$_i$ et devra choisir une autre destination.

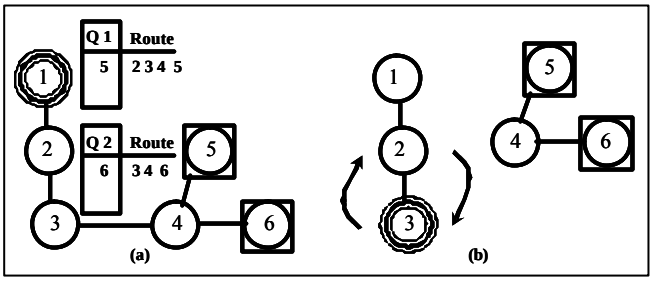


Figure 3. *Circulation en boucle du jeton.*

– Pourquoi le message ***LinkInfo*** transporte les informations d'une requête?

A partir du moment qu'un nœud peut être détenteur du jeton et peut se déconnecter momentanément et sachant qu'il peut être détenteur du jeton alors il serait intéressant de lui envoyer une requête pour réclamer le jeton.

5. Conclusion

Dans cet article, nous avons présenté une solution au problème d'EM dans les réseaux mobiles ad hoc. Le protocole présenté se base sur les liens physiques, n'utilise aucune structure logique. Les requêtes sont acheminées suivant un rayon dynamique de diffusion déterminé selon la connaissance locale afin de couvrir une proportion de nœuds demandeurs. Le jeton est géré selon une méthode qui combine la circulation et la demande. De plus, la politique de satisfaction des nœuds favorise en même temps les nœuds les plus proches et les requêtes les plus anciennes. La mobilité des nœuds fait partie intégrante de la solution proposée grâce aux différents mécanismes présentés dans ce qui précédait. Aussi, le protocole tolère les arrivées, départs et pannes de nœuds mais sans la perte de jeton. Par conséquence, le protocole favorise la scalabilité du réseau. Cependant, il ne tolère que des partitionnements temporaires. La comparaison de performances avec d'autres protocoles nécessite une étude de performances qui nécessite une simulation vu que le rayon de diffusion est dynamique (,dépendant de la charge de demande dans le réseau et du facteur temps).

Remerciement

Nous tenons à remercier Monsieur A. Lahmar pour sa collaboration quant à la réalisation de ce travail.

Bibliographie

Badrinath B. R., Acharya A., and Imielinski T., '"Structuring distributed algorithms for mobile hosts", Proc. of the 14th Intern. Conf. on Distr. Comp., pp. 21 – 28, 1994.

Baldoni R., Virgillito A., Petrassi R., "A distributed mutual exclusion algorithm for mobile ad-hoc networks", Proc. of the Seventh Inter. Symposium on Computers and Communications, pp. 539 – 544, 1 - 4 July 2002.

Benchaïba M., Bouabdallah A., Badache N., Ahmed-Nacer M., "Distributed mutual exclusion algorithms in mobile ad hoc networks : an overview", ACM SIGOPS Operating Systems Review, Vol. 38 , Issue 1, pp. 74 – 89, Jan. 2004.

Carvalho O., Roucairol G., "On mutual exclusion in computer networks", Com. of the ACM, Vol. 26, N° 2, pp. 146 - 147, Feb. 1983

Chang Y., Singhal M., and Liu M., "A fault tolerant algorithm for distributed mutual exclusion", In Proc. of 9th IEEE Symposium On Reliable Dist. Systems, pp. 146 - 154, 1990.

Dhamdhere D. M. and Kulkarni S. S., "A token based k-resilient mutual exclusion algorithm for distributed systems", Information Processing Letters, vol. 50 N° 3, pp. 151-157, 1994.

Gafni E. and Bertsekas D., "Distributed algorithms for generating loop-free routes in networks with frequently changing topology.", IEEE Transactions on Communications, C- 29(1), pp. 11 – 18, 1981.

Lamport L., "Time, clocks, and the ordering of events in a distributed system.", Communications of the ACM, Vol. 21, N°7, pp. 558 - 565, July 1978.

Le Lann G., "Distributed systems, towards a formal approach", IFIP Congress, Toronto, pp. 155 - 160, 1977.

Maekawa M., "A √N algorithm for mutual exclusion in decentralized systems", ACM Trans. on Comp. Systems, Vol. 3, N°2, pp. 145-159, May 1985.

Malpani N., Chen Yu, Vaidya N. H., Welch J. L., "Distributed token circulation in mobile ad hoc networks", IEEE Transactions on Mobile Computing, Volume 4, Issue 2, pp. 154 - 165, March-April 2005.

Raymond K., "A tree-based algorithm for distributed mutual exclusion", ACM Transactions on Computer Systems, Vol. 7 N° 1, pp. 61 - 77, Feb. 1989.

Ricart G. and Agrawala A. K., "An optimal algorithm for mutual exclusion in computer networks", Comm. of the ACM, Vol. 24, N° 1, pp. 9 - 17, Jan. 1981.

Ricart G., Agrawala A. K., "Author response to 'on mutual exclusion in computer networks', by Carvalho and Roucairol", Communication of the ACM, vol. 26, N°2, pp.147-148, Feb. 1983.

Singhal M., "A heuristically-aided algorithm for mutual exclusion in distributed systems", IEEE Trans. On Computers, Vol. 38, N°5, pp. 651-662, May 1989.

Singhal M., ''A taxonomy of distributed mutual exclusion", Journal of Parallel and Distributed Computing, Vol. 18, N°1, pp. 94-101, 1993.

Suzuki I., Kazami T., "A distributed mutual exclusion algorithm. ACM Transactions Computer Systems", Vol. 3, N° 4, pp. 344 - 349, 1985.

Walter J., Welch J., Vaidya N., "Mutual Exclusion Algorithm for Ad Hoc Mobile Networks.", Wireless Net., Vol. 9, No. 6, pp.585 - 600, Nov. 2001.

Annexe : Primitives et routines du protocole d'exclusion mutuelle

```
###########              Initialisation
Init_node()
begin
if (i = 0)
    then {
        Holder_i := True;  Token_State := ONRoute;Token_Route := {};
        Token_Req_Set := {}; Token_NS := {}; Token_Last_Holder := 0;
        Token_Locked := False;
} else {  Holder_i := False;
}end_if;
NS_i := 0; Request_NS_i := 0; State_i := Idle; Nb_attempts_i :=0;  Last_Holder_i := -1;
R_i := -1; N_i := {}; Q_i := {}; Ignored_Neighbors_i := {}; Received_NS_i := {};
aware_i := {};
end
```

Les primitives:

```
###########              Demander l'accès à la SC
Event Ask_For_CS
begin
    State_i := Requesting; NS_i := NS_i + 1; Request_NS_i := NS_i;  Nb_attempts_i := 1;
    if (Holder_i) then {  Enter_CS( ) ;          /* le jeton est au repos */
    } else {                              /* calculer le rayon de diffusion */
       R_i := Calculate_Radius( );     Route_i := i; aware_i := N_i ∪ i;
       for each (j ∈ N_i) do{send Request(i, NS_i, Route_i, R_i, Nb_attempts_i, aware_i)to j;
       }done ; Set (RequestTimer_i);  /* déclencher timer demande de i */
   }end_if
end
##########      Recevoir une requête
Event Receive_Request: Request(j, NS_j, Route_j, R_j, Nb_attempts_j, aware_j)
bool insert := False;
begin
     insert := AddRequest(j, NS_j, Route_j, Nb_attempts_j);/* insérer dans la file */
     if (insert) then {
          NS_i := max(NS_i, NS_j);
          if (Holder_i) then {          /* l'inscrire dans le jeton */
                    Token_Req_Set := Token_Req_Set ∪ j;
                    Token_NS := Token_NS + new_element(j); Token_NS(j) := NS_j;
          }end_if;
          if ((!Holder_i) or (Holder_i and State_i = InCS)) then {
                    if (R_j = -1) then { R_j := Calculate_Radius( ); /* rayon indefini */
                    } else {  R_j := R_j -1;
                    }end_if;
                    if (R_j > 0) then { Route_j := Route_j + i; /* suivre la diffusion de la
                                                                  requête de j */
```

```
                    for each (k ∈ (N_i - aware_j)) do {send Request(j, NS_j, Route_j,
                                                        Nb_attemps_j, Ni ∪ aware_j) to k;
                    }done
                  }end_if
          } else { VehicleToken();/* passer le jeton au prochain */
          }end_if
    }end_if
end
###########                    Recevoir le jeton
Event Receive_Token : Token(Token_State, Token_Route, Token_Req_Set,
                            Token_NS, Token_Last_Holder, Token_Locked)
begin
   Holder_i := True; UpdateQueues(); NS_i := max(NS_i, max(Token_NS));
   if(State_i = Idle) then {  Token_NS(i) := NS_i + 1; }end_if
   if (Token_State = ONRoute) then {          /* le jeton est sur la route */
         Last_Holder_i := Token_Last_Holder; Ignored_Neighbors_i := {};
         Token_Last_Holder := i;
         if ( (State_i = Requesting) and [(last(Token_Route) = i) or(!Token_Locked)])
               then { if (last(Token_Route) = i) then { Token_Route := {};}end_if;
                  Enter_CS( );                  /* entrer en SC */
         } else { VehicleToken( );
         }end_if
    } else {        /* Token_State = NORoute */
       Ignored_Neighbors_i := Ignored_Neighbors_i ∪ Token_Last_Holder;
       Token_Last_Holder := i;
       if (State_i = Requesting) then { Enter_CS( ); } else { VehicleToken( ); }end_if
    }end_if
end
###########                    Quitter la SC
Event Leave_CS
begin
    State_i := Idle;
    if (Holder_i) then {
         Token_Req_Set := Token_Req_Set - i; Token_NS(i) := NS_i + 1;
         if (!Empty(Q_i)) then { VehicleToken( ); }end_if
    }end_if
end
###########                    Fin de ValidTimer
Event End_Timer ValidTimer_i(j)
begin
     Q_i(j).Valid := False;
end
###########                    Fin de RequestTimer
Event End_Timer RequestTimer(i)
begin
      Nb_attempts_i := Nb_attempts_i + 1; aware_i := N_i ∪ i;
```

```
      if (Nb_attempts_i modulo Request_Threshold)
         then { R_i := Calculate_Radius( ); Route_i := i;
                for each (j ∈ N_i) do { send Request(i, Request_NS_i, Route_i, R_i,
                                                    Nb_attempts_i, aware_i) to j;
                }done; Set(RequestTimer_i);
       } else { State:= Unable; /* Etat d'échec */
      }end_if
end
###########                    Lien rompu
Event Link_Down( i, j)
begin
        N_i := N_i - j;
end
###########                    Lien établi
Event Link_Up( i, j)
begin
     N_i := N_i ∪ j;
     if (State_i=requesting) then { send Link_Info( i, Request_NS_i,  Nb_attempts_i) to j
    }end_if
end
####                        Recevoir Link_info
Event Receive Link_Info(j,ns_j, NB_attempts_j)
bool insert := False;
begin
    insert := AddRequest(j, ns_j, j, Nb_attempts_j);/* insérer dans la file */
     if (insert) then {   NS_i := max(NS_i, ns_j);
        if (Holder_i) then {          /* l'inscrire dans le jeton */
              Token_Req_Set := Token_Req_Set ∪ j;
              Token_NS := Token_NS + new_element(j); Token_NS(j) := ns_j;
              if ( State_i != InCS)) then {VehicleToken }end_if
        }end_if
    } end_if
end
                                  Les routines
#######################              Calculer rayon
integer Calculate_Radius( )
list local_list := {}; integer R := -1, pos;
function Nb_Hops, Nb_elements; /* Supposés définies
begin
   for each (j ∈ Q_i) do {
        if (Q_i(j).Valid) then { //insertion avec tri
              local_list := inset(local_list, Nb_Hops(Q_i(j).Route));
        }end_if
    }done;
    if(!Empty(local_list)) then { pos := round(P * Nb_elements(local_list));
                                  R := local_list[pos];
```

```
    }enf_if; return R;
end
########################          Entrer en SC
Enter_CS( )
begin
    State_i := InCS ; NB_attemps_i := 1;
End
########################          Donner jeton au prochain
VehicleToken( )
bool neighbor := False, Retreived := False, break := False; integer dest := -1;
begin
    if (!Empty(Token_Route)) then { /* faire suivre le jeton sur sa route */
        if (Last(Token_Route) ∈ Ni) then { neighbor := True;
                                          Token_Route := Last(Token_Route);
        } else { Token_Route := Token_Route - i;
                 neighbor := First(Token_Route) ∈ N_i;
        }end_if
        if (!neighbor) then { Retreived := Retreive_Route(Token_Route); } end_if;
        if(neighbor or (!neighbor and Retreived)) then {
          Holder_i := False;  send Token(Token_State, Token_Route,
          Token_Req_Set, Token_NS, Token_Last_Holder, Token_Locked)
                                                    to First(Token_Route);
        }enf_if
   }end_if;
   if (Holder_i) then { /* choisir un autre nœud destinataire*/
        for each (j ∈ Q_i) do { if (j ∈ N_i) then { Q_i(j).Route := j; Q_i(j).Valid := True;
                                                    Set(ValidTimer(j)); }end_if
        }done; Sort(Q_i);
        for each ((j ∈ Q_i) and !break) do {
           neighbor := First((Q_i(j).Route ∈ (N_i –Ignored_Neighbors_i));
           if (!neighbor) then { Retreived := Retreive_Route(Q_i(j).Route); } end_if;
           if (neighbor or(!neighbor and Retreived)) then {
                Token_Route := Q_i(j).Route; Token_State := ONRoute;
                Token_Last_Holder := i;
                if (Q_i(j).Nb_attempts > Lock_Limit) then {Token_Locked := True;
                } else { Token_Locked := False;
                }end_if; Holder_i := False;
                send Token(, Token_State, Token_Route,   Token_Req_Set,
                          Token_NS,  Token_Last_Holder, Token_Locked)
                                                    to First(Q_i(j).Route);
                 break := True;
           }enf_if
        }done
  }end_if;
  if (Holder_i) then { /* le jeton sera routé avec NORoute pour rechercher un nouveau
                                                    destinataire*/
```

```
      if (Last_Holder_i ∈ N_i) then { dest := Last_Holder_i; Last_Holder_i := -1;
      } else{
         if (Empty(N_i -Ignored_Neighbors_i)) then {Ignored_Neighbors_i := {};}end_if;
         dest := k ∈ (N_i - Ignored_Neighbors_i) / NS of k is MAX
      }end_if;
      Token_State := NORoute; Token_Route := {};  Token_Locked := False;
      if (dest != -1) then {
          send Token(Token_State, Token_Route, Token_Req_Set,  Token_NS,
                                    Token_Last_Holder, Token_Locked) to dest;
      }end_if
   }end_if
end
#######################          Récupérer route
bool Retreive_Route(Route)       /* l'argument Route est entré par adresse */
set_j := {}; bool break := False;
begin
   for each ((j := Last(Route); j != First(Route); j := Previous(j, Route)) and !break)
         do { if (j ∈ (N_i - Last(Ignored_Neighbors_i))) then {
                  set_j := set_j + j; Route := Reverse( set_j ); break := True;
              } else {  set_j := set_j + j;
              }end_if
         }done; return break;
end
#######################                    Ajouter demande
bool AddRequest(j, NS_j, Route_j, Nb_attempts_j)
bool insert := False;
begin
 if (j ∈ Q_i) then {
  if ((NS_j > Q_i(j).NS) or [(NS_j = Q_i(j).NS) and (Nb_attempts_j > Q_i(j).Nb_attempts )])
     then { Q_i(j).NS := NS_j;  Q_i(j).Nb_attempts := Nb_attempts_j;
     Q_i(j).Route := Route_j; Q_i(j).Valid := True; Set(ValidTimer(j)); insert = True;
      Sort(Q_i);
  } else { if ((NS_j = Q_i.(j).NS) and (Nb_attempts_j = Q_i(j).Nb_attempts)) then {
              if (Nb_Hops(Route_j) < Nb_Hops(Q_i(j))) then { Q_i(j).Route := Route_j;
                  Q_i(j).Valid := True; Set(ValidTimer(j)); Sort(Q_i); insert :=  True;
              }end_if
            }end_if
   }end_if
} else { if (j ∈ Received_NS_i) then {
          if (Received_NS_i(j) < NS_j) then { Received_NS_i(j) := NS_j;
                  CreateNewElement(j, Route_j, NS_j, True, Nb_attempts_j);
                  Set(ValidTimer(j)); insert := True; Sort(Q_i);
           } end_if
          }else { Received_NS_i := Received_NS_i + new_elementNS (j);
                  Received_NS_i(j) := NS_j:
                  CreateNewElement(j, Route_j, NS_j, True, Nb_attempts_j);
```

```
                    Set(ValidTimer(j)); insert := True; Sort(Qi);
          }end_if
}end_if;  return insert;
######################                              UpdateQeueus
UpdateQueues()
begin
    for each (j ∈ Qi) do {
       if (Exists(Token_NS(j)) then {
           if (Qi(j).NS < Token_NS(j)) then { /* maj de Qi  */
                    Delete(Qi(j)); Desactivate(ValidTimer(j));
           } else { /* (Qi(j).NS >= Token_NS(j)) donc maj jeton */
                    Token_Req_Set := Token_Req_Set ∪ j;  Token_NS(j) := Qi(j).NS;
            } end_if
       } else {Token_NS := Token_NS + new_element(j); Token_NS(j):= Qi(j).NS;
       } end_if
     }done;
      for each (j ∈ Token_Req_Set) do {       /* maj de Qi       */
          if (j ∉ Qi(j)) then { AddRequest(j, Token_NS(j),NULL , 1); }end_if
       }done
end
```

SESSION 4

LES NOUVELLES ARCHITECTURES DE GESTION

Un middleware flexible et scalable pour la gestion intégrée à large échelle

Mehdi Kessis[1], Pascal Dechamboux[1], Claudia Roncancio[2], Alexandre Lefebvre[1], Thierry Coupaye[1]
[1]France Télécom R&D, MAPS/AMS
28 chemin du Vieux Chêne, 38243 Meylan, France
{mehdi.kessis, pascal.dechamboux, alexandre.lefebvre}@rd.francetelcom.com
Caudia.ronconcio@imag.fr
LSR-IMAG

RESUME

Aujourd'hui, les réseaux et les services font partie du paysage quotidien de l'entreprise et de nos foyers. Avec les bienfaits de ces nouvelles technologies de l'information et de la communication sont apparus leurs méfaits. Ces environnements sont de plus en plus sophistiqués, répartis, hétérogènes. Les infrastructures de gestion classiques montrent leurs limites face à ces nouvelles problématiques. L'administration (supervision, installation, configuration, réparation, etc.) de ces infrastructures devient une tâche très ardue. De nouvelles architectures de gestion deviennent indispensables pour appréhender cette complexité. Une approche possible de traiter ces problèmes est de concevoir des systèmes de gestion flexibles et autonomes. Ce papier se situe au cœur de cette problématique de gestion. Il expose les nouveaux défis à relever et propose une architecture de gestion de type middleware, à la fois flexible, scalable et autonome pour la gestion intégrée des réseaux et des services.

Mots-clés: Gestion intégrée, autonomie architectures réparties, composants, gestion des flux, gestion autonome

1. Introduction

Les deux dernières décennies ont été marquées par un essor remarquable de l'informatique répartie (réseaux paire à paire, les grilles de calcul, l'informatique hypervasive, etc.) et des services de télécommunication. Désormais, les nouvelles technologies de l'information et de la télécommunication envahissent aussi bien les entreprises que les foyers et font partie intégrante de notre vie quotidienne. Vingt cinq ans après la naissance des ordinateurs personnels, l'ITU[1] a recensé, en 2002[2], plus d'un milliard d'ordinateurs dans le monde. Selon le même document, ce chiffre va doubler en 2008 (en seulement 6 ans). Il est évident que le nombre des équipements évoluent à un rythme exponentiel. Aujourd'hui une entreprise gère, en moyenne, une dizaine voir des milliers d'équipements. Un opérateur ou fournisseur de service, quant à lui, il gère des milliers voir des millions d'équipements[3]. L'opérateur France Télécom gère, par exemple, un parc de plus d'un million de passerelles de services Livebox[4].
Cette complexité, sans cesse croissante, met en exergue la nécessité de disposer de nouvelles approches facilitant l'administration de ces ressources à large échelle.
Ce papier se situe au cœur du contexte introduit ci-dessus. Il traite la problématique de gestion de réseaux et des ressources hétérogènes répartis et déployés en grand nombre (à large échelle). Nous proposons une architecture de gestion de type middleware. Le but de cet article est de présenter l'architecture de ce middleware d'administration proposé. La suite de l'article expose les défis à relever dans un contexte de gestion des systèmes répartis, hétérogènes et déployé en grands nombres (à large échelle). Nous détaillons, après, la solution de gestion à base de middlewares flexibles. Nous discutons, à la fin du papier, l'originalité de l'approche proposée, tout en la positionnant par rapport aux différents travaux existants.

[1] International Telecommunication Union URL http://www.itu.int/home/
[2] sources: ITU, Gartner Dataquest, IDC, 02-03
[3] The international Engineering Consortium(http://www.iec.org), Performance Management of Next Generation Networks
[4] http://www.silicon.fr/getarticle.asp?ID=11906

2. *Evolution des besoins de l'administration*

Les réseaux et des services sont des entités dynamiques [25]. Elles évoluent constamment et requièrent, par conséquent, de nouvelles techniques et de nouvelles approches pour les gérer de manière efficace. En étudiant l'état de l'art des infrastructures d'administration des systèmes et des réseaux nous avons identifiés plusieurs défis à relever dans le cadre de la gestion des systèmes répartis et hétérogènes; (*i*) le problème de passage à l'échelle, (*ii*) le problème de la gestion des ressources hétérogènes, (*iii*) le problème de l'automatisation des infrastructures de gestion et (*iv*) le problème de la flexibilité de ces infrastructures pour faire face aux changements de l'état des ressources gérées et l'évolution des besoins en terme d'administration. Nous étudions ces problèmes plus en détails dans cette section.

2.1. *Le passage à l'échelle*

Le passage à l'échelle est un problème critique dans le monde l'administration. Néanmoins, aucune définition formelle de la scalabilité n'est disponible dans la littérature [9]. Nous retiendrons cette définition dans le cadre de ce papier *"Un système est dit scalable s'il maintient constant la dégradation de ces performances lorsque sa taille évolue"* [8]. Les premières architectures d'administration, proposées vers la fin des années 80, étaient principalement centralisées. Les systèmes de supervision des réseaux IP, proposés à l'époque étaient principalement basés sur la première version du protocole SNMP [5]. Quelques années plu tard, ce modèle a montré ses limites quant au passage à l'échelle [20]. Dès lors, des efforts ont été déployés afin de proposer de nouveaux modèles répartis permettant de pallier ces limites. Une conséquence de ces travaux est la naissance de plusieurs approches de gestion réparties [21] ; la gestion répartie avec un grand frère (big brother) centralisé [22], la gestion répartie hiérarchique [23], la gestion par délégation [27], la gestion répartie via des agents mobiles [24]. Bien que différents, tous ces travaux préconisent la répartition de l'intelligence et des traitements afin de contourner le goulet d'étranglement, lié à un administrateur unique (modèle centralisé), lors du passage à l'échelle. Chacune de ces approches est applicable dans un contexte de gestion particulier [3]. Généralement, on opte pour l'une ou l'autre. Toutefois, basculer de l'une d'entre elles à une autre est souvent très coûteux et requière assez souvent la remise en question de toute l'infrastructure de gestion sou jacente. La mise en place de canevas modulaires, flexibles et à base de composants est une piste intéressante qui permet d'appliquer l'une ou l'autre de ces solutions en fonction de l'évolution des ressources à gérer.

2.2. *L'hétérogénéité*

De plus en plus d'équipements et de ressources hétérogènes cohabitent dans les mêmes réseaux. La cause de la défaillance d'un service par exemple, peut être liée à la défaillance d'un équipement ou du réseau. Afin d'identifier rapidement ces interdépendances, il est indispensable de représenter ces ressources et leurs interactions. Nous estimons que disposer d'informations de différents niveaux (réseau, services, équipements) permet de comprendre d'éventuels phénomènes (intrusion, panne, etc.) difficiles à expliquer sans ce type d'approche. Cette discipline d'administration est désignée sous le nom d'administration intégrée [12]. L'objectif est d'intégrer tous les types de gestion (gestion des applications, des équipements, etc.) dans une plateforme unique au lieu d'avoir une plateforme pour chacun des aspects à administrer. Ce type d'infrastructures permet de disposer d'une vue plus complète des systèmes à administrer. Une piste possible pour disposer de ce type de vue consiste à construire un référentiel d'informations de gestion basé sur le modèle d'information standard CIM (Common Information Model) [4].

2.3. La Flexibilité

La flexibilité d'une infrastructure de gestion est sa capacité à d'étendre et de reconfigurer ses composants afin de répondre à des nouveaux besoins ou de s'adapter à de nouvelles situations. Rares sont les travaux qui on traité la flexibilité des infrastructures de gestion. Goldszmidt [27] a proposé une infrastructure de gestion par délégation basée sur la notion d'élasticité (la capacité d'entendre les fonctionnalités d'une entité de gestion à qui on délègue de nouvelles responsabilités d'administration). Actuellement, la plupart des infrastructures de gestion existantes sont généralement des applications monolithiques. Ces applications ne sont ni assez flexibles ni extensibles. Une piste possible pour pallier cette faiblesse, consiste à bâtir une nouvelle génération des systèmes de gestion basée sur des infrastructures à base de composants logiciels reconfigurables.

2.4. L'autonomie

Une infrastructure autonome est une infrastructure capable d'auto optimiser ses performances, d'auto-corriger ses pannes, de s'auto-protéger et de s'auto-configurer [14]. Dans le cadre d'administration des systèmes répartis à large échelle, l'administrateur risque d'être confronté à un volume assez important d"informations et d'évènements à gérer. L'automatisation des réponses suite à l'occurrence de ces évènements devient alors indispensable. Afin d'automatiser le processus d'administration, l'infrastructure de gestion doit disposer de moyens d'observation, de décision et de contrôle. L'autonomie est une propriété indispensable dans le cadre de gestion des systèmes répartis à large échelle. En effet, automatiser certains traitements réduit le coût de gestion et accroît son efficacité (plus de réactivité). Assez souvent, un administrateur est plus vulnérable qu'un système automatisé à base de règles face à une cascade d'alarmes.

3. Technologies avancées pour l'administration répartie à large échelle

Afin de relever les défis soulignés dans cette section, nous proposons d'explorer les pistes identifiées dans la section suivante.

3.1. Une architecture à base de composants

Nous estimons qu'une infrastructure de gestion scalable doit être avant tout flexible. Afin de concevoir une application flexible, nous nous sommes appuyés sur le modèle à composants Fractal [1]. Ce modèle offre la possibilité de concevoir, de (re)configurer dynamiquement des infrastructures à base de composants. Une infrastructure Fractal est composée d'un ensemble de composants Fractal interconnectés et inter reliés.
Chaque composant Fractal est composé d'une membrane et d'un contenu. Les composants Fractal peuvent être imbriqués de façon hiérarchique, d'où les notions de composants et de composants composites. La Figure.1 illustre les différents éléments qui forment un composant Fractal et un exemple d'imbrication de composants Fractal.

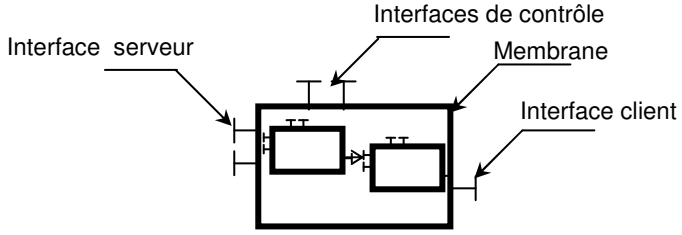


FIGURE 1 Architecture d'un composant Fractal

Les composants Fractal communiquent avec ses semblables via des interfaces. Les interfaces dans Fractal, ont un rôle central. Il existe principalement deux catégories distinctes d'interfaces : (a) les interfaces métier; ces interfaces constituent les points d'accès externes au composant et (b) les interfaces de contrôle (ou contrôleurs) qui prennent en charge des propriétés non fonctionnelles du composant. Fractal met à disposition un ensemble d'interfaces de contrôle qui sont la gestion du cycle de vie (*life cycle controler*), des liaisons (*bind controler*), des attributs (*attribute controler*), du contenu (*content controler*) pour les composites. Le modèle étant extensible, il est possible de définir ses propres interfaces de contrôle.

3.2. La technologie Fractal-JMX

Java Management eXtension (JMX) [26] est le standard d'instrumentation dans le monde Java. La philosophie de la technologie JMX consiste à exposer des interfaces des classes et des instances java à superviser via un serveur MBean. Les interfaces sont exposées sous la forme de composants java nommés Mbean (Management Bean). Toutes les implémentations des serveurs MBean proposent un service de supervision permettant de définir des alarmes, des seuils des gauges sur les MBeans enregistrés.
Des travaux récents ont appliqué ce standard dans le cadre de la gestion des composants Fractal. La technologie Fractal-JMX[5] est le fruit de ces travaux. Cette technologie permet d'automatiser la gestion des environnements fractalisés (conçu en Fractal). Elle permet d'introspecter ces environnements et d'exposer les informations concernant les différentes entités observables. Le principe consiste à injecter dans l'application des agents en Fractal qui encapsulent, entre autre, un serveur MBean. La Figure.2 illustre un agent Fractal JMX à l'intérieur d'un composite Fractal. L'agent nous offre une vue sur l'ensemble des composants imbriqués au sein du composant observé.

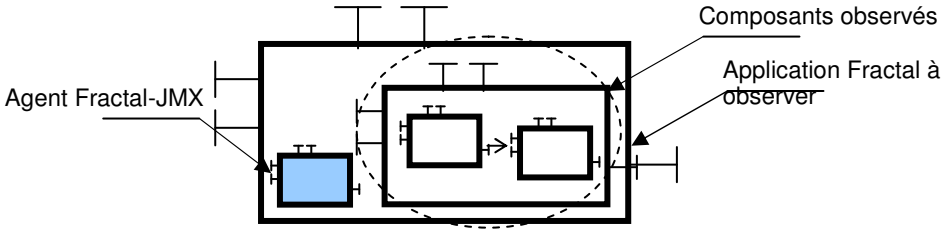


FIGURE 2 Observation des composants Fractal via un agent Fractal-JMX

3.3. Un Modèle d'information unifié

Le modèle d'information de gestion est un élément clef de toute solution d'administration de réseaux et de services [19]. Celui-ci définit la vue qu'offrent les équipements, les réseaux et les services aux applications de supervision [2]. Le modèle d'information que nous avons adopté dans le cadre de ce travail est le modèle CIM (Common Information Model) [4].

CIM est un standard DMTF[6]. Il représente un modèle orienté objet qui permet d'unifier la représentation et la description des ressources à administrer. Il permet de modéliser aussi bien les ressources logiques (application, composants, services, politiques, bases de données, etc.) que les ressources physiques (routeur, capteur, PC, etc.). CIM se base sur des techniques de structuration et de conception selon un paradigme orientée objet unifié (encapsulation, héritage, etc.).

[5] http://Fractal.objectweb.org/tutorials/jmx
[6] DMTF : Distributed Management Task Force, http://www.dmtf.org/home

4. *Un middleware de gestion autonome, flexible et scalable pour la gestion intégrée*

4.1. *Architecture générale*

Nous proposons dans cette section l'architecture globale d'un middleware flexible et autonome pour l'administration intégrée des réseaux et des systèmes. Le canevas proposé est baptisé Zeus. Le middleware Zeus est vu par les applications de gestion comme de boite noire. Zeus se base sur l'ensemble des technologies présentées dans les sections précédentes pour construire des réseaux logiques (*overlay networks*) de nœuds assurant la fonction de gestion pour un ensemble d'applications de gestion. La structure des nœuds et leurs interconnections sont gérés de façon transparente par Zeus et l'administrateur du middleware. Le réseau logique formé est totalement flexible, reconfigurable et extensible. Les nœuds et les liens entre ces nœuds sont également reconfigurables. Ainsi, les connexions entre les nœuds sont des connexions virtuelles, indépendantes du réseau physique sous-jacent. Il est important que les infrastructures de gestion flexibles soient des overlay networks, car cela permet aux pairs de s'abstraire du réseau de communication, et ainsi permet de fédérer plusieurs réseaux différents (IP, ad-hoc, etc.) entre eux. Ainsi, même si un des réseaux sous-jacents tombe en panne, le réseau logique restera viable après la perte des nœuds qui étaient dans cette partie du réseau physique. La Figure.3 illustre l'architecture globale de l'infrastructure d'administration proposée.

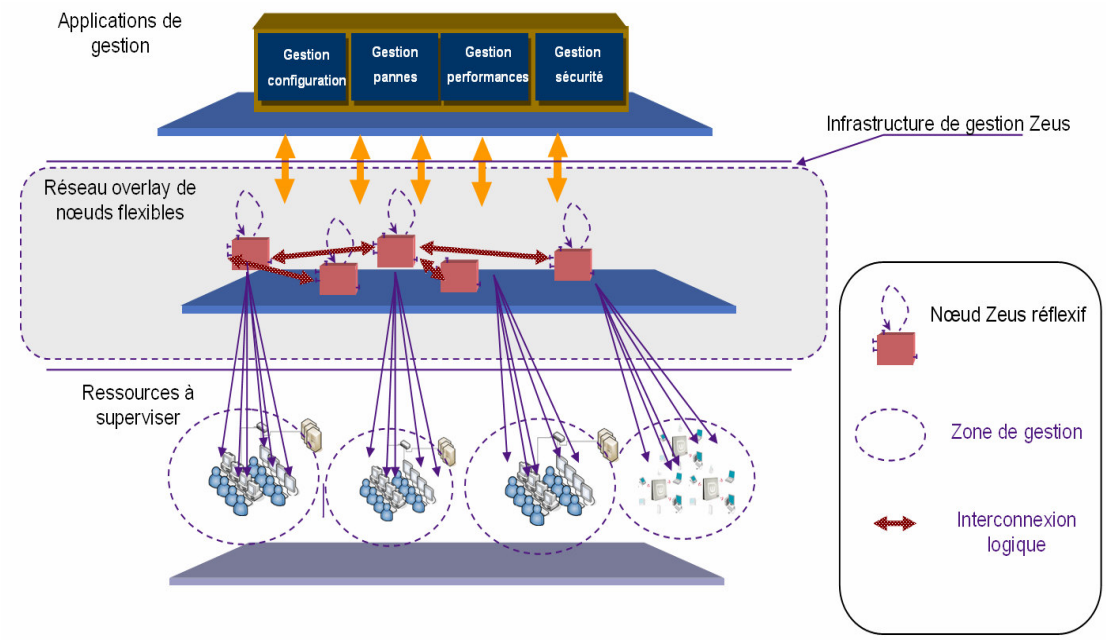


FIGURE 3 Architecture globale des middlewares Zeus

Cette figure illustre l'architecture globale d'une infrastructure Zeus. Les applications de gestion sollicitent les nœuds Zeus afin d'effectuer leurs tâches d'administration. Les nœuds Zeus collaborent afin de répondre à ces sollicitations. Ils jouent le rôle de médiateur entre les ressources gérées et les applications de gestion. Chaque nœud Zeus offre un ensemble de services (gestion de référentiel, localisation des ressources, localisation des nœuds Zeus, etc.). Les services Zeus concernent les aspects non fonctionnels de l'administration. Les applications de gestion qui sollicitent Zeus, lui délèguent ces fonctions.
Les nœuds Zeus sont capables de s'auto-observer grâce à des agents Fractal-JMX qu'ils embarquent. Ces nœuds sont également capables de s'auto-optimiser et de s'auto-réparer (via les interfaces de contrôle Fractal et les mécanismes de reconfiguration dynamique supportés par Fractal. La figure 4 illustre un exemple d'architecture interne d'un nœud Zeus.

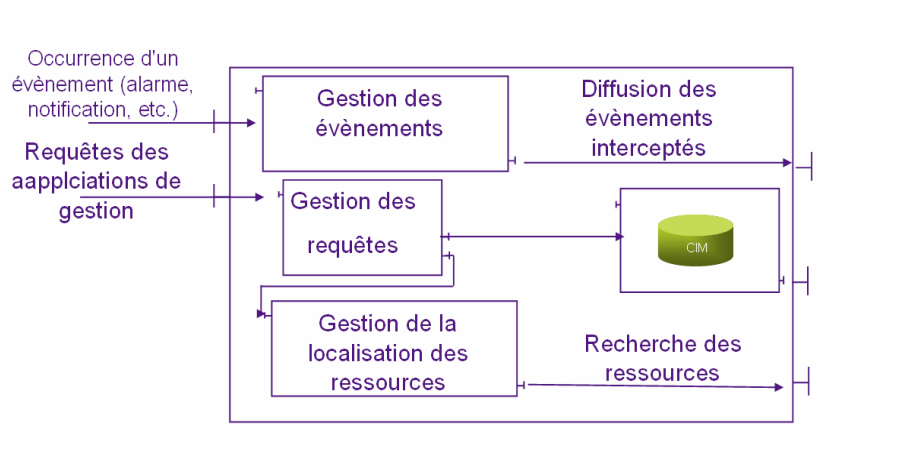


FIGURE 4 Vue interne d'un nœud Zeus

Zeus est basé sur deux API; (*i*) une API pour la construction et le paramétrage du middleware et (*ii*) une API pour l'exploitation du middleware conformément à la JSR 48 [6].

L'API d'administration de Zeus permet d'instancier, d'assembler le réseau de nœuds Zeus. C'est grâce à cette API que l'administrateur de Zeus peut déployer et modeler des middlewares de gestion répartis. Cette API offre globalement deux types d'opérations: (i) des opérations de gestion de cycle de vie des nœuds (addZeusNode, DeleteZeusNode, UpdateZeusNode, etc.) et des opérations sur les services offerts par ces nœuds (AddService, removeService, UpdateService, etc.).Une fois déployé, le middleware offre des services aux applications de gestion qui le sollicitent via la deuxième API proposée (API d'exploitation).

La deuxième API est conforme à l'API client du JSR 48 [6]. Les applications clientes peuvent manipuler des référentiels CIM gérés par les nœuds Zeus. Ceci étant, Zeus se comporte comme un CIM Object Manager [6].

4.2. La boucle gestion autonome

Nous avons présenté, dans les sections précédentes, le modèle à composant Fractal et la technologie Fractal-JMX pour l'instrumentation de ces composants. Nous étudions dans cette section notre approche de gestion autonome basée sur la combinaison de ces deux technologies. Lors du déploiement de Zeus, un overlay de nœuds Zeus est formé. L'overlay est constitué de nœuds Fractalisés. Ces nœuds sont observables via l'agent Fractal JMX. Ils sont contrôlables via les interfaces de contrôle Fractal. Les nœuds Zeus font partie d'une infrastructure Zeus (un composant à gros grain). Nous injectons un agent Fractal JMX au sein de ce composant afin d'observer le comportement des composants (nœuds Zeus et leurs composants (services)). Grace ces agents nous collectons des informations et nous interceptons des évènements (alarmes, notifications, etc.). Les actions à entreprendre, suite à la remontés des alertes et des notifications, sont spécifiés au niveau de ces bases sous formes de règles évènements action simples. La figure 5 illustre l'exemple d'un agent Fractal JMX qui observe un nœud de l'overlay de nœuds Zeus. L'agent Fractal-JMX communique au service de gestion de règle de gestion les informations interceptées. Ce dernier peut agir sur l'état des composants concernés par l'action à entreprendre. Dans cette figure deux actions sont à pourvoir; (a) une action sur un lien enter deux nœuds Zeus et (b) une action concernant un nœud Zeus.

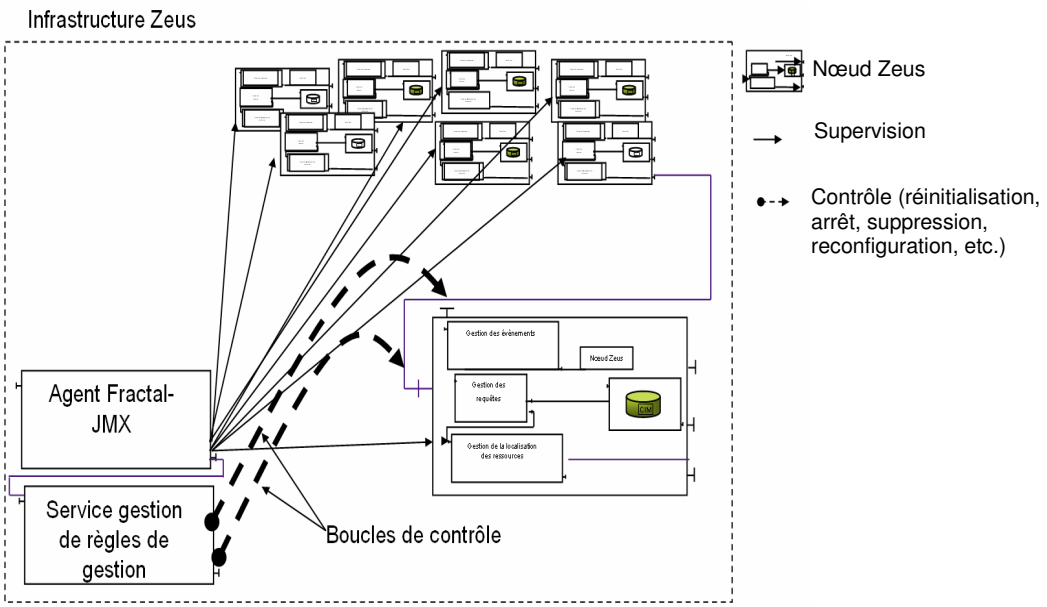


FIGURE 5 Gestion autonome des nœuds Zeus

5. *Discussion*

Plusieurs avantages découlent de l'adoption des architectures à base de composants dans le cadre de l'administration des réseaux et des services. V. Wadel et al. [17] a vivement recommandé l'adoption de l'approche à base de composants dans la conception des solutions d'administration dans le monde de la télécommunication. Baude et al. [18] a proposé une architecture Fractalisée pour la gestion des grappes de calcul. JMX [26] propose un service de supervision sous forme de composant MBean. Le premier apport de ce type d'infrastructure est la flexibilité.

Les applications de gestion sont conçues indépendamment de la taille du réseau et du nombre des propriétés à gérer. C'est le middleware Zeus qui gère le routage des requêtes, la persistance des données, la localisation des ressources, etc. Ses nœuds logiques lui dotent d'une indépendance vis-à-vis des ressources physiques allouées. Ainsi donc, un nœud Zeus peut être déployé aussi bien sur un capteur que sur grappe de calcul. Ce qui fait de notre infrastructure à la fois une infrastructure flexible et scalable.

La flexibilité de Zeus réside également dans la capacité de "*pluguer"* des composants. L'architecture interne du middelware Zeus n'est pas figée. On peut ainsi, modeler l'overlay, à la carte, des nœuds en fonction des besoins de gestion. Les services sont construits à partir d'un assemblage de composants Fractal. Ces composants ainsi que leurs interactions (bindings) sont totalement reconfigurables.

Une autre originalité de Zeus consiste à proposer une infrastructure répartie pour la gestion des référentiels CIM. Nous adoptons dans le cadre de ce travail des API de type JDO[7] [16] pour gérer la persistance de ces référentiels CIM.

6. *Travaux connexes*

A notre connaissance aucun travail n'a exploré l'application des middlewares flexibles dans le cadre de la gestion intégrée des réseaux et des services.

L'architecture CIM/WBEM [6] est une architecture standard basée sur le modèle CIM comme modèle d'information, XML comme langage de représentation des informations, et http comme protocole de transport des informations. L'architecture CIM/WBEM se décline e

[7] JDO est une spécification définissant le concept de 'persistance transparente' pour un objet Java vers une source de données transactionnelle.

quatre couches; (a) *les applications d'administration* (l'ensemble des applications clientes qui sollicitent les services du serveur CIM/WBEM), (b) *le CIMOM (CIM Object Manager)* (composant cœur de l'infrastructure WBEM, il assurer l'interaction entre les applications d'administration et les fournisseurs de l'information de gestion) (c) *les fournisseurs (*communiquent directement avec les ressources gérées et capturent les informations et les évènements provenant de ces ressources), (d) *le référentiel (*une sorte de base de données qui regroupe les informations de gestion).

Zeus fournit des fonctions similaires aux services des CIMOM WBEM. L'avantage de ZEUS par rapport aux implémentations WBEM existantes (WBEMServices [8], OpenWbem[9], Pegasus[10], Microsoft WMI[11]) sont sa flexibilité et sa scalabilité. Tous ces CIMOM sont centralisés et monolithiques. Ce genre d'infrastructure est applicable dans le cadre de la gestion d'un réseau local. Toutefois, il ne passe pas l'échelle.

Marvel [10] est un environnement distribué de gestion des informations d'administration base sur la technologie Web. L'architecture de Marvel se décline en une Hiérarchie de serveurs Marvel interconnectés à un ensemble d'agents. Marvel donne aux administrateurs systèmes la possibilité de définir des vues et des vues agrégées sur les ressources administrées et sur des groupes de ressources administrées. Il offre ainsi, un mécanisme de navigation pour interroger et superviser l'ensemble de mini bases de données éparpillées sur le réseau. Les serveurs Marvel calculent en permanence (de façon synchrone et asynchrone) des vues sur les informations de gestion. Les vues Marvel suivent un modèle orientée objet. Marvel est une infrastructure monolithique et qui n'est pas flexible ni autonome. Son modèle d'information est similaire au modèle standard OSI. Toutefois, Il reste un modèle propriétaire non extensible.

Baude et al [18] propose une infrastructure d'administration répartie à base de composants Fractal et d'agents mobiles pour les grappes de calcul [25]. L'infrastructure proposée se base sur la technologie Proactive. Il s'agit d'une technologie basée sur des agents mobiles qui peuvent être Fractalisés. L'architecture proposée dans ce travail est flexible. Toutefois, le modèle d'information proposé est ad hoc.

Weaver Query System (WQS) est un système qui permet de construire des vues globale sur le trafic dans les réseaux. Pour ce faire, il construit une sorte de réseau overlay au dessus du réseau physique, à travers lesquels vont circuler les informations de gestion. Ce travail se base sur le concept de patrons de navigation dans les réseaux ou les systèmes répartis. Ces patrons désignent des comportements typiques identifiés lors des échanges d'informations entre des systèmes répartis. Ces comportements peuvent représentés par des algorithmes de parcours de graphes. WQL [28] st le langage proposé pour exprimer des requêtes envoyé au serveur WQS. Ce travail s'est focalisé sur les algorithmes de diffusion et les patrons de navigation dans un réseau. Il a négligé les aspects flexibilité, modèle d'information et autonomie.

Astrolabe [29] est un service d'administration distribué scalable. Astrolabe permet la supervision (collecte, diffusion et agrégation des informations) du changement d'état d'un ensemble de collections de ressources réparties. Pour ce faire, il se base sur des mécanismes de groupement en zones hiérarchiques, un protocole pair à pair de diffusion des informations et des techniques probabilistes pour la gestion des duplications et de la cohérence des données. Chaque zone contient un ensemble d'agents qui calculent en permanence des fonctions d'agrégation spécifiées par les applications qui sollicitent les services Astrolabe (applications d'administration, applications de localisation des ressources, etc.). La création des zones se fait implicitement lors de la définition et de la création des agents. Ces derniers créent les zones s'ils n'existent pas. À chaque zone est associée une liste de propriétés qui constituent sa MIB. Chaque zone dispose d'un ensemble de fonctions d'agrégations

[8] http://wbemservices.sourceforge.net
[9] http://openwbem.sourceforge.net
[10] http://www.openpegasus.org
[11] http://laurent-dardenne.developpez.com/articles/wmi-p1

qui calculent les valeurs des attributs des MIB des zones. Une fonction d'agrégation est une sorte de requête SQL qui prend comme paramètre la liste des MIB des zones filles et qui produit un résumé de leurs attributs. Les zones feuilles constituent une exception. Elles contiennent des zones filles virtuelles. Ces zones sont locales aux agents et sont accessibles en écriture et non générées par des fonctions d'agrégation. Chaque agent communique avec les autres agents via le protocole pair à pair. Astrolabe a été conçu avec l'hypothèse de supporter des *MIB* de taille moyenne (quelques centaines ou milliers d'octets). L'agent local de chaque zone calcul les valeurs de sa MIB locale. Il récupère des informations sur les MIB de ses voisins via un protocole de diffusion des informations. Ce travail mise sur la scalabilité mais néglige l'aspect gestion autonomique et ne traite pas la question de l'hétérogénéité des ressources.

7. *Conclusion et perspectives*

Au terme de ce travail, nous avons abordé la problématique de gestion autonome des systèmes hétérogènes et répartis déployés large échelle. Le point focal de ce travail était la flexibilité des infrastructures de gestion face aux changements fréquents des besoins d'administration. Ce ci étant, nous avons proposé un canevas de gestion intégré des systèmes hétérogènes et répartis à large échelle. Le canevas proposé est baptisé Zeus. Nous nous sommes appuyés sur le modèle à composants Fractal pour la conception et le développement de Zeus. Zeus consiste un overlay de nœuds logiques est déployé suite à l'initiative de l'administrateur afin de fourni ces services. Ces nœuds gèrent en autre des référentiels CIM représentant ainsi l'ensemble des ressources gérées (physiques et logiques) de façon standardisée. Les nœuds collaborent afin d'offrir des services (filtrage, agréation, routage, gestion de la persistance, etc.) aux applications de gestion qui sollicitent Zeus. L'infrastructure de gestion Zeus est autonome. En effet, Zeus offre la possibilité d'observer l'état de ces nœuds via des agents Fractal-JMX. Un ensemble de politique de gestion est prédéfini en réponse aux évènements remontés par ces agents. Les actions peuvent être appliquées via les interfaces de contrôle des nœuds Zeus.
Zeus est actuellement en cours de développement. Plusieurs améliorations sont en cours d'étude. Nous envisageons de développer d'avantage sa base de règles et de politiques de gestion. Nous étudions également la possibilité d'embarquer un service de gestion de flux de donnés au niveau des agents Zeus afin de prendre en considération les flux d'informations remontées par les agents et les sondes. Cette extension peut être particulièrement utile pour la gestion du trafic sur le réseau ou pour la supervision temps réel des informations fluctuantes. Nous envisageons également de doter les agents Zeus d'une certaine forme de mobilité en proposant des agents fatalisés et mobiles grâce à la technologie ProActive[12].

Bibliographie

[1] E. Bruneton, T. Coupaye, and J.B. Stefani. "*Recursive and Dynamic Software Composition with Sharing"*. Seventh International Workshop on Component-Oriented Programming (WCOP02), Malaga, Spain, June 10-14, 2002.

[2] Olivier Festor, *"Ingénierie de la gestion de réseaux et de services : du modèle OSI à la technologie active"*, mémoire d'habilitation à diriger la recherche, Université Henry Poincaré Nancy 1, décembre 2001.

[3] Jean-Philippe Martin-Flatin, *"Web-Based Management of IP Networks and Systems"*, Edition Wiley, Wiley Series in communications Networking & Distributed Systems, September 2002.

[4] Common Information Model Standard, URL: http://www.dmtf.org/standards/cim/

[5] *"Simple Network Management Protocol (SNMP) "*Network Working Group, RFC 1157.

[12] http://www-sop.inria.fr/oasis/ProActive/

[6] Java Specification Request N°48 (JSR 48): WBEM Services Specification, URL: http://www.jcp.org/en/jsr/detail?id=48
[7] Won-Ki Hong, J. Kim, J., Park, J.: *"A CORBA-Based Quality-of-Service Management Framework for Distributed Multimedia Services and Applications"*, DSOM98
[8] R. V. Renesse, K. P. Birman, and W. Vogels, *"Astrolabe: A Robust and Scalable Technology for Distributed System Monitoring, Management, and Data Mining"*, ACM Transactions on Computer Systems, Vol. 21, No. 2, May 2003, Pages 164–206.
[9] Mark D. Hill, *"What is scalability?"*, ACM SIGARCH Computer Architecture News, December 1990.
[10] N. Anerousis and G. Hjálmtysson, *"View-based Management of Services in a Programmable Internetwork"*, in Proceedings of the 2000 Network Operations and Management Symposium, Honolulu, HI, April 2000.
[11] W. Sattallings, *"SNMP, SNMP V2, SNMP v3 and RMON 1 and 2"*, Third Edition, Addison-Wesley, April 2004.
[12] H.G. Hegering, S. Abeck and B. Neumair. *"Integrated Management of Networked Systems: Concepts, Architectures, and their Operational Application"*, San Francisco, CA, USA, 1999.
[14] Jeffrey O. Kephart and David M. Chess." The vision of autonomic computing", IEEE Computer, 36(1):41–50, January 2003.
[15] CIM Query Language Specification, SQP0202, DMTF: URL: http://www.dmtf.org/standards/published_documents/DSP0202.pdf
[16] *"Java™Data Objects (JDO) specification"*, URL: http://java.sun.com/products/jdo/
[17] V.Wade, D.Lewis, C.Malbon, T.Richardson, L.Sorensen and C Stathopoulos, *"Component Integration Technologies for Telecoms Management Systems"*, TCD-CS, Technical Report, Trinity College Dublin Computer Science Department, 1999.
[18] F. Baude, D. Caromel and M. Morel "*From Distributed Objects to Hierarchical Grid Components*", International Symposium on Distributed Objects and Applications (DOA), Catania, Sicily, Italy, November 2003.
[19] Jean-Philippe Martin-Flatin, *"Toward Universal Information Models in Enterprise Management"*, Proc. VLDB 2001 Workshop on Databases in Telecommunications (DBTel 2001), Rome, Italy, September 2001.
[20] T. M. Chen and Stephen S. Liu,*"A Model and Evaluation of Distributed Network Management Approaches"*, IEEE Journal on selected areas in communications, Vol. 20, N°. 4, may 2002.
[21] Mohsen Kahani, H.W. Peter Beadle, *"Decentralized Approaches for Network Management"*, SIGCOMM, July '1997
[22] Young-pa So and Edmund H.Durfee. *"Distributed Big Brother"*. In *Proceedings of the Eighth IEEE Conference on Artificial Intelligence Applications*, pages 295-301, March 1992.
[23] M.R. Siegl, G. Trausmuth Technical University of Wien, Austria, *"Hierarchical network management, a concept and its prototype in SNMPv2"*, Article, 10 pages, May 1995.
[24] Huy Hoang To, Shonali Krishnaswamy, Bala Srinivasan, *"Mobile agents for network management: when and when not!"*, Symposium on Applied Computing, Proceedings of the 2005, Pages: 47 - 53 , Santa Fe, New Mexico, 2005.
[25] N. Angloumine, O. Cherkaoui, "*Pratique de la gestion de réseau Solutions de contrôle et de supervision d'équipements réseau pour les entreprises et les opérateurs télécoms*", édition Eyrolles, mars 2003.
[26] Sun microsystems, Java Extension Management (JMX), URL: http://java.sun.com/products/JavaManagement/
[27] GOLDSZMIDT, German *" Distributed Management by Delegation"*. 1996. Ph.D Thesis – Graduate, School of Arts and Sciences, Columbia University, New York.
[28] K.-S. Lim and R. Stadler, "Real-time Views of Network, Traffic using Decentralized Management", Proc. 9th International Symposium on Integrated Network Management, 15-19 May 2005, Nice, France.

[29] R. Vna Renesse, K. P. Birman, and W. Vogels, *"Astrolabe: A Robust and Scalable Technology for Distributed System Monitoring, Management, and Data Mining"*, ACM Transactions on Computer Systems, Vol. 21, No. 2, May 2003, Pages 164–206.

Architecture de gestion de passerelles domestiques de services

Yvan Royon
Stéphane Frénot

INRIA Ares / Laboratoire CITI, INSA Lyon
Bâtiment Léonard de Vinci
21 avenue Capelle
69621 Villeurbanne cedex
{yvan.royon, stephane.frenot}@insa-lyon.fr

RÉSUMÉ *Nous voyons aujourd'hui une popularisation des services Internet amenés au domicile des usagers (données, téléphonie et télévision sur IP). Demain, les services offerts vont exploser en nombre et en diversité : domotique, santé, coexistence de plusieurs fournisseurs de télévision… Nous pensons qu'un nouveau modèle économique est sur le point d'apparaître. Il s'agit d'intégrer des flux multimédia et des applications venant de plusieurs fournisseurs, en les hébergeant sur un environnement unique : la passerelle domestique, remplaçante du traditionnel modem.*
Nous proposons une architecture de gestion pour ce modèle économique ouvert, dit multi-services et multi-fournisseurs. Nous nous penchons notamment sur la gestion locale à la passerelle, ou encore sur la répartition des responsabilités en cas de panne. Notre implémentation est basée sur OSGi et JMX.

ABSTRACT. *Today we see a popularization of services brought to connected homes, i.e. date, television over IP and voice over IP. Tomorrow, services should grow in both number and diversity: domotics, health care, competition between TV providers… We believe in the birth of a new business model, which is to integrate multimedia streams and applications from several providers, by hosting them on a single physical environment: the residential gateway, a replacement for the usual modem.*
We propose a management architecture for such an open, multi-service, multi-provider business model. We study in particular the management infrastructure on the gateway, and the distribution of responsibilities in case of malfunction. Our implementation is based on OSGi for the gateway and JMX for management.

MOTS-CLÉS : *plate-forme de services, gestion, OSGi, JMX.*

KEYWORDS: *open service gateways, management, OSGi, JMX.*

1. Introduction : modèle économique

1.1. *Hier et aujourd'hui*

Durant ces dernières années, le taux d'équipement des ménages en matière d'accès Internet a explosé.

Hier, l'offre commerciale proposée aux particuliers était limitée à la connectivité IP. Ceci permettait d'obtenir un ensemble de services de base, non garantis (best effort) : web, e-mail, newsgroups, etc.

Aujourd'hui, les fournisseurs d'accès Internet (FAI) étendent leur offre en proposant ce que l'on nomme le *Triple Play* : données, voix, vidéo. En plus des services Internet traditionnels, des services à valeur ajoutée (téléphonie sur IP, télévision sur IP) assurent une meilleure attractivité aux FAI.

Cependant, ce modèle économique reste fortement limité. Les FAI contrôlent la chaîne de livraison des services, de bout en bout. L'offre de services reste donc fermée, avec une concurrence limitée et très peu de diversité.

1.2. *Demain*

1.2.1. *Multi-Play*

L'évolution envisagée, communément nommée *Multi-Play*, est d'ouvrir la concurrence sur les offres multimédia. Il s'agit pour l'usager de pouvoir contracter plusieurs abonnements de téléphonie et/ou de télévision et de choisir dynamiquement quel fournisseur de services utiliser. En d'autres termes, il s'agit de casser le monopole des fournisseurs d'accès Internet sur la chaîne de provision de services chez l'usager. Toujours dans le but d'être plus attractif, les propositions de *Multi-Play* intègrent généralement la gestion de la qualité de service de bout en bout, du fournisseur de services jusque chez l'usager.

1.2.2. *Multi-services*

Une autre évolution à ce modèle économique, complémentaire à la précédente, est d'ouvrir également l'environnement d'exécution des services. L'idée la plus communément répandue pour faire face au *Multi-Play* est que chaque fournisseur de services propose son propre terminal, ou *set-top box*, à l'usager client. Ceci mène à une explosion des équipements à la maison, la multiplication de leurs coûts de fabrication et d'entretien, et une plus grande complexité dans le réseau domestique. Pour éviter cela, nous soutenons l'idée qu'un équipement unique peut héberger et faire coexister l'ensemble de ces services. Cet équipement, la passerelle domestique, doit permettre de transformer les modems et autres FreeBox que nous connaissons en plates-formes de services.

Mais un environnement d'exécution de services ouvert dispose d'un avantage encore plus décisif. Il permet d'ouvrir une infinité de possibilités quant à la gamme

de services offerte. Ainsi, il devient aisé de proposer des services de domotique, de santé, d'aide aux personnes âgées, ou encore de loisirs divers. Les fournisseurs de services peuvent alors être des fabricants d'électroménager, des hôpitaux, etc.

1.3. *Acteurs en présence*

Nous pouvons identifier plusieurs acteurs autour de ce modèle économique :

- L'opérateur réseau fournit la connectivité IP à la passerelle domestique.
- Le fournisseur de passerelles produit la passerelle domestique.
- Le gestionnaire de passerelles s'assure du bon fonctionnement de la passerelle et de la connectivité. Il fournit l'infrastructure et les ressources pour que les services de chaque fournisseur puissent s'exécuter dans de bonnes conditions.
- Le fournisseur de services vend un service particulier, qui peut être une application, un flux multimédia, ou encore un gestionnaire d'équipement pour la maison.
- Le gestionnaire de services gère les services d'un fournisseur particulier, vérifie leur état et leurs paramètres.

Ces rôles peuvent bien entendu être cumulés. Ainsi, il est probable que le FAI assume l'identité de l'opérateur réseau, du fournisseur de passerelles et du gestionnaire de passerelles. De même, les fournisseurs de services en seront souvent aussi les gestionnaires.

La section 2 ci-dessous décrit les contraintes qui découlent du modèle économique multi-fournisseurs, multi-services présenté, ainsi que les éléments à mettre en œuvre pour y répondre. La section 3 présente notre implémentation de ces éléments. La section 4 aborde des travaux connexes, tandis que la section 5 conclut l'article en donnant des pistes de travaux futurs.

2. Mise en oeuvre

Pour réaliser cette architecture multi-play, multi-fournisseurs et multi-services, plusieurs éléments sont nécessaires. Premièrement, à l'extrémité de la chaîne, chez l'usager, il faut un environnement d'exécution capable de faire coexister et coopérer des applications de diverses origines. Deuxièmement, une architecture de gestion est nécessaire, d'une part pour que chaque fournisseur puisse administrer ses services, d'autre part pour répartir les responsabilités en cas de panne ou mauvais fonctionnement.

2.1. *Plate-forme d'exécution*

L'usager dispose à son domicile d'un équipement unique le reliant au monde extérieur : la passerelle domestique. Il s'agit à la fois d'un modem (qui fournit la connectique) et d'un environnement d'exécution pour les services (flux et applicatifs), comme représenté en figure 1. Ainsi, sur un même environnement, plusieurs tiers peuvent déployer et gérer des services. Ceci offre un certain nombre d'avantages, tout en répondant à des contraintes du modèle économique.

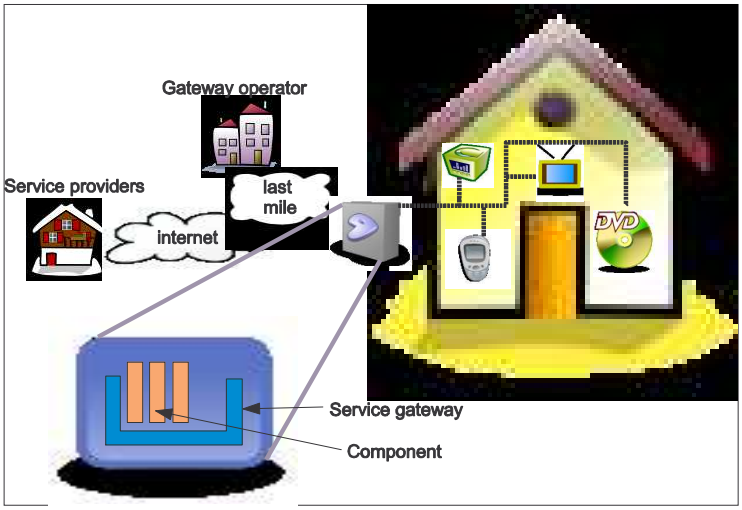


Figure 1. *Principe d'une passerelle de services ouverte*

2.1.1. *Avantages d'un environnement d'exécution ouvert*

L'environnement d'exécution de la passerelle domestique peut être un simple système d'exploitation où un ensemble d'applications s'exécutent, sans se connaître. Mais cet environnement unique prend tout son intérêt s'il est orienté service, au sens génie logiciel du terme (SOA – Service-Oriented Architecture). Dans ce cas, des développeurs isolés peuvent, sans connaître leurs confrères ni le code qu'ils produisent, faire en sorte que toutes ces applications soient interopérables.

Par exemple, dans le cas d'applications multimédia, un premier fournisseur peut fournir des codecs. D'autres doivent pouvoir fournir des lecteurs vidéo, des lecteurs de flux audio, des abonnements à ces flux, *etc*, tout en réutilisant ces codecs. Mieux, les codecs peuvent être améliorés et mis à jour « à chaud », en cours de fonctionnement, en perturbant peu voire pas du tout les services qui les utilisent, et surtout sans redémarrer la passerelle domestique. Autre avantage, plusieurs

implémentations des codecs peuvent cohabiter : l'environnement d'exécution fournit alors à l'application demandeuse le codec le plus adapté.

En plus de la réutilisation des services, la réutilisation du matériel et de l'environnement est un énorme avantage. Ainsi, si un hôpital veut surveiller un pacemaker, il n'a pas à fabriquer, vendre et livrer sa propre *set-top box* : il lui suffit d'écrire l'applicatif et de le déployer sur la passerelle domestique.

2.1.2. *Contraintes*

Nous avons vu que, sur un même point d'accès qui relie un réseau domestique à un réseau opérateur, plusieurs tiers peuvent déployer et gérer des services. Il y a donc partage de ressources. Idéalement, les services provenant de fournisseurs différents ne devraient pas interférer : il devrait donc y avoir une isolation des ressources (CPU, mémoire, disque) allouées. Mais surtout, chacun doit pouvoir accéder à distance à ses propres services, sans toucher ni même « voir » les services de fournisseurs potentiellement concurrents.

Que chaque fournisseur puisse gérer ses propres services sous-entend qu'une architecture de gestion de bout en bout est présente.

2.2. *Modèle de gestion*

De l'environnement multi-fournisseurs présenté ci-dessus naissent plusieurs impératifs. Tout d'abord, si un service ne fonctionne pas chez l'usager, plusieurs éléments peuvent en être la cause. Il faut savoir la diagnostiquer afin de répartir les responsabilités. Ensuite, nous constatons qu'un mécanisme de délégation de la gestion est déjà présent : la passerelle domestique concentre les informations de gestion des équipements de la maison. Il faut encore concentrer les informations de gestion d'un ensemble de passerelles domestiques, pour arriver jusqu'au fournisseur de services. Nous devons donc définir ces différentes délégations de la gestion.

2.2.1. *Répartir les responsabilités*

Prenons l'exemple d'un usager abonné à flux audio. L'usager n'entend pas le flux promis. La faute peut venir de plusieurs niveaux :

- La source d'émission. Le responsable est le fournisseur du flux.
- Le réseau qui transporte le flux. Le responsable est l'opérateur réseau. (Nous simplifions ici le cas d'opérateurs réseau multiples, du dernier kilomètre, etc.)
- L'application lectrice du flux. Le responsable est le fournisseur de service, par le biais du développeur.
- Une mauvaise manipulation de l'utilisateur.

Il doit exister une fonction de gestion capable de désigner le fautif. Elle permettra de différencier les bons des mauvais fournisseurs, de services comme d'accès réseau. Elle permettra aussi d'envoyer la facture à qui de droit.

2.2.2. *Délégation de la gestion et passage à l'échelle*

Chez l'usager, l'environnement domestique héberge plusieurs équipements, tels qu'un frigo connecté, une cafetière intelligente, ou une brosse à dents électronique. Tous ces appareils sont surveillés et gérés par des applications dédiées, hébergées sur la passerelle domestique.

A l'opposé de la chaîne, le fournisseur de services dispose d'un outil de supervision, capable de remonter les informations et alarmes en provenance de ses propres services déployés chez ses clients usagers. Le cas échéant, le même outil peut inférer sur la configuration desdits services. Dans le présent article, cette inférence est manuelle, par hypothèse simplificatrice.

Selon le modèle économique décrit, un fournisseur de services dispose d'une base potentielle de clientèle énorme, à savoir tous les utilisateurs d'Internet à domicile (soit un tiers de la population Française en 2004). Il est alors évident que se pose le problème de passage à l'échelle. Tout comme la passerelle domestique fédère les données de gestion des équipements de la maison, il est préférable de placer des intermédiaires, ou concentrateurs, entre le fournisseur de services et l'usager.

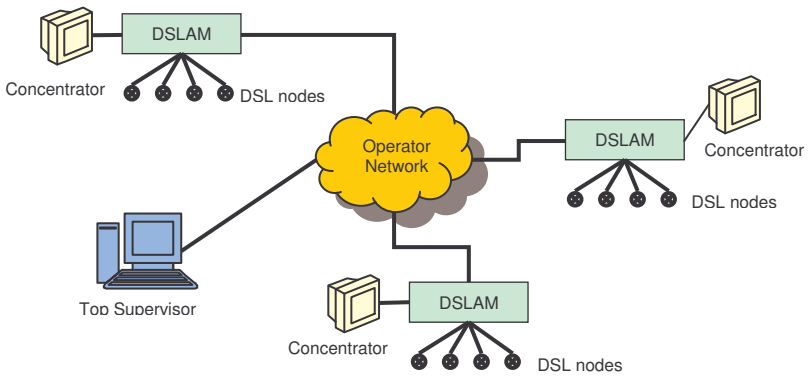


Figure 2. *Chaîne de gestion des services : du superviseur aux passerelles*

Puisque nous nous penchons sur le cas de l'accès Internet aux domiciles, l'emplacement de ces concentrateurs est tout indiqué. Le DSLAM, ou autre équipement similaire, fédère déjà la connectique entre quelques centaines de boucles

locales et le réseau opérateur. Il pourrait donc abriter un élément intelligent, qui d'un côté concentre les informations des passerelles domestiques des usagers, et de l'autre côté filtre les informations à remonter au superviseur (figure 2).

3. Implémentation

En partant d'un modèle économique, nous avons présenté une architecture supportant la gestion et l'exécution de services dans un environnement multi-fournisseurs. L'implémentation que nous proposons se base sur et étend les spécifications OSGi, présentées ci-dessous, pour l'environnement d'exécution de la passerelle domestique. L'architecture de gestion est, elle, basée sur JMX.

3.1. *Introduction à OSGi*

OSGi (OSGi Alliance, 2005) est une proposition pour définir de façon standard la manière de gérer à distance des services et des périphériques matériels utilisés dans un environnement local. La spécification OSGi définit les API nécessaires pour pouvoir exécuter et gérer des services sur une passerelle. L'API OSGi repose sur la machine virtuelle et le langage Java. Elle spécifie les trois concepts suivants :

- **Le conteneur de services :** c'est un démon qui garantit l'exécution des différents composants hébergés. Il permet l'association entre des clients demandant l'accès à des services, et des composants implantant ces services. En résumé son rôle est d'enregistrer des services et de gérer localement l'activité de la plate-forme.

- **Les services standards :** la spécification définit un certain nombre de services standards (http, log, user admin…) que la plate-forme propose. Ces services sont directement exploitables par d'autres services. Ainsi un nouveau service déployé peut toujours faire appel à un service web pour fournir une interface web.

- **Un modèle de composants simple :** le modèle à composants d'OSGi est volontairement simple et centralisé (l'administration est distante, mais l'exécution des composants se fait de manière centrale sur la passerelle). Le modèle repose sur le concept de *bundle* comme unité de transport et de déploiement de services. C'est une archive Java (.jar) qui peut être installée à distance sur la passerelle. La description du contenu du *bundle* se fait par un fichier de description au format Manifest Java (couples "clé: valeur"). Un *bundle* OSGi peut contenir des services offerts, des *packages* Java et des librairies natives chargées en fonction de l'environnement d'exécution hôte.

Du point de vue application, le conteneur de services gère le cycle de vie des composants. Il vérifie que les composants sont autorisés à s'installer, à s'exécuter et à exploiter les services fournis par d'autres composants. Si un composant exportant des *packages* est stoppé, tous les composants qui en dépendent sont stoppés automatiquement. Les composants de la plate-forme OSGi répondent à un cycle de

vie précis, qui est géré par le conteneur de services: un composant peut être installé, résolu (tous les composants dont il dépend sont démarrés), démarré ou arrêté.

Il existe plusieurs implantations d'OSGi venant aussi bien d'industriels, comme IBM SMF (IBM, 2003), que du monde open-source, comme Felix (Apache Software Foundation, 2005).

3.2. *Multi-fournisseurs dans OSGi*

Sur l'environnement unique qu'est la passerelle domestique, plusieurs fournisseurs peuvent déployer des services. Ces fournisseurs peuvent être indépendants et même concurrents. Il faut donc s'assurer que les services des uns soient isolés des services des autres. Pour formaliser cette isolation, nous avons proposé la notion de plate-forme de services virtuelle. La passerelle domestique héberge une plate-forme de services principale, opérée par le gestionnaire de passerelle. Celle-ci héberge et gère des plates-formes de services virtuelles, chacune appartenant à un fournisseur de services distinct, comme décrit dans la figure 3. Le fournisseur de services peut alors gérer tous les services s'exécutant dans sa propre plate-forme virtuelle, comme s'il s'agissait d'un environnement OSGi standard.

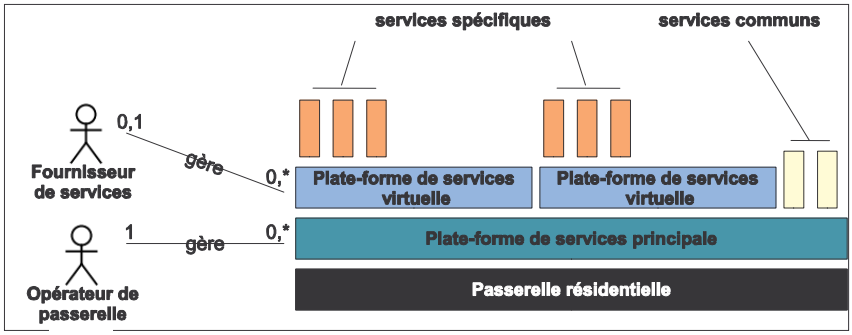


Figure 3. *Passerelle domestique multi-services, multi-fournisseurs*

La plate-forme de services principale se rapproche de la notion d'utilisateur root Unix, et les plates-formes virtuelles de celle d'utilisateurs standard.

3.3. *Gestion dans OSGi*

Les spécifications OSGi restent très génériques sur l'architecture de gestion. Ni architecture ni protocole ne sont indiqués. En outre, il n'y a pas d'éléments de facturation ni de gestion de la performance.

3.3.1. *Protocoles*

Nous n'imposons pas non plus de protocole particulier entre la passerelle domestique et les équipements de la maison. En effet, chaque fournisseur est libre d'utiliser un protocole propriétaire ou un protocole standardisé (UPnP, Jini…), pour peu qu'il fournisse un *bundle* OSGi capable de comprendre ce protocole.

Sur le réseau distant par contre, nous utilisons JMX, pour l'agent présent sur la passerelle domestique ainsi que pour les échanges. L'intégration de JMX dans OSGi a fait l'objet de précédents travaux (Fleury, 2003).

3.3.2. *Modèle*

Comme présenté plus haut, nous avons ajouté la notion de plate-forme virtuelle, ainsi que son lien avec son gestionnaire. Le modèle d'acteurs autour de la gestion proposé par OSGi devient alors le suivant, notre seul ajout étant la flèche courbe qui représente le multi-fournisseurs (ici multi-gestionnaires) :

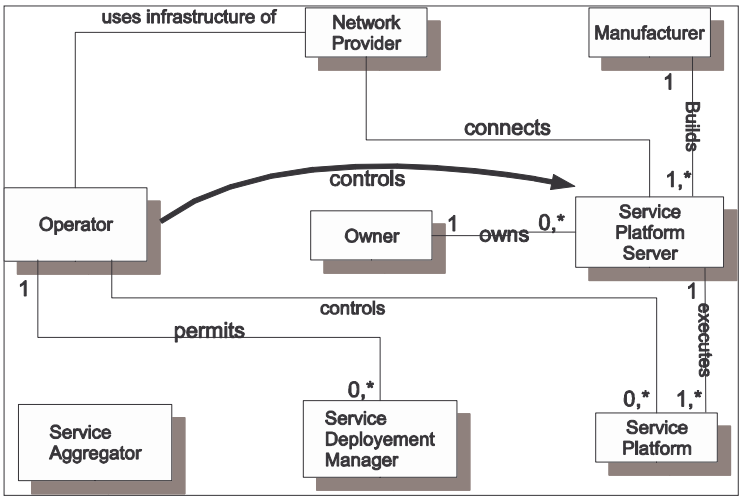


Figure 4. *Amendement du modèle de gestion distante dans OSGi*

3.3.3. *Interfaces de gestion*

Nous proposons deux familles d'interfaces de gestion. La première (tableau 1) donne au gestionnaire de passerelle la possibilité de créer et supprimer une plate-forme virtuelle (un environnement dédié à un fournisseur de services particulier). Elle permet également de vérifier l'utilisation de ressources de chaque fournisseur, de vérifier ses droits d'exécution, et de prendre mesure de l'utilisation d'un service pour facturation. La deuxième famille d'interfaces (tableau 2) est destinée aux fournisseurs, et leur permet de gérer le cycle de vie de leurs propres services.

```
public interface LifeCycleMgmt{
  public void startSubGw(String provId) throws Exception;
  public void stopSubGw(String provId) throws Exception;
}

public interface PerformanceMgmt{
  public Ticks getCPUUsage(String providerId);
  public Memory getMemoryUsage(String providerId);
  public BandWidth getBandWidthUsage(String providerId);
  public boolean isAlive(String providerId);
}

public interface SecurityMgmt{
  public void setCredential(String providerId,
                            Credential newCred);
  public void challengeCredential(String providerId,
                                  Credential testCred);
}

public interface AccountingMgmt{
  public String notify(Event event);
  public Ticks getServiceUsage(String providerId,
                               String serviceClazz);
```

Tableau 1. *Interfaces de gestion pour gestionnaire de passerelle principale*

```
public interface BundleLifeCycleMgmt{
  public void startBundle(String serviceId) throws Exception;
  public void stopBundle(String serviceId) throws Exception;
  public Collection listBundles();
  public boolean isAlive(String serviceId);
}
```

Tableau 2. *Interface de gestion pour fournisseur de services*

3.4. *Outils*

Afin de vérifier que l'implantation de l'architecture de gestion fonctionne bien, nous avons développé plusieurs outils.

3.4.1. *Console de supervision*

Le premier outil est une console de supervision (JMXConsole), qui tourne elle-même au-dessus d'OSGi. Elle découvre dynamiquement les services gérables sur une passerelle particulière. La gestion de l'environnement ou d'un service particulier peut nécessiter une interface graphique spécifique. Celle-ci est découverte automatiquement par la console, via un mécanisme de plug-ins.

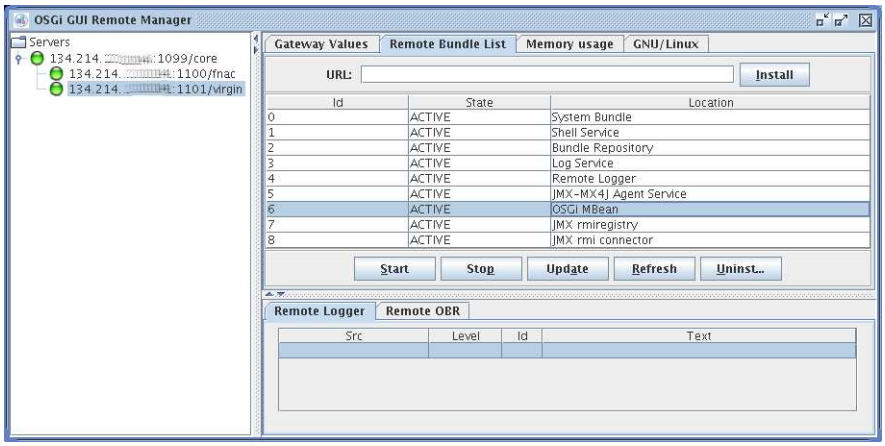


Figure 5. *Console de supervision*

3.4.2. *Répartition des responsabilités*

Le deuxième outil proposé (OPing) est un ping multi-niveaux. Il détermine si la connectivité IP jusqu'à une passerelle donnée fonctionne, puis si la passerelle elle-même est à l'écoute, et enfin si un service particulier tourne bien.

4. Travaux connexes

Le modèle économique présenté ici est partiellement à l'étude dans plusieurs projets européens. En particulier, le *multi-play* est l'élément moteur du projet IST MUSE (Projet MUSE, 2004). Son rôle est de proposer pour un futur proche des réseaux d'accès et de bordure ouverts sur le *multi-play*, le tout à bas prix pour les citoyens européens. D'autres projets comme Amigo ou MediaNet proposent des travaux connexes. Amigo vise à définir un réseau ambiant intelligent, intégré et facile d'utilisation pour le domicile. MediaNet est tourné vers les communications multimédia et la chaîne de distribution de ces flux.

Sur des points plus précis de la présente proposition, comme la notion d'isolation des fournisseurs de services dans un même environnement, d'autres travaux sont à noter. Le fer de lance de l'isolation d'applications dans le monde Java est l'ancien projet Barcelona de Sun. La machine virtuelle multi-tâches (Czajkowski, 2003), toujours en développement, offre une véritable isolation de ressources (CPU, mémoire). La notion d'approche orientée services, fondamentale dans notre cas, n'est cependant pas dans son cahier des charges. D'autres proposent à l'inverse d'ajouter dans OSGi un contrôle de ressources CPU des services présents (Yamasaki, 2005).

5. Conclusion et travaux futurs

Nous avons explicité un modèle économique où plusieurs tiers peuvent fournir, par le biais d'un unique équipement d'interconnexion, des services au domicile de l'usager. Ces services peuvent être des applications, des abonnements à des flux multimédia, des gestionnaires pour les équipements connectés dans la maison, ou toute autre prestation imaginable.

Nous avons présenté une implantation de la passerelle domestique multi-fournisseurs, basée sur OSGi. Nous avons également proposé une implantation de l'architecture de gestion à distance des services, construite autour de JMX.

D'autres travaux restent à mener pour compléter cette proposition. En premier lieu, il nous faut spécifier plus en détail, puis implémenter, la notion de concentrateur dans ce modèle économique. Egalement, nous travaillons sur les aspects sécurité liés à la problématique multi-fournisseurs, suivant deux axes. D'une part, le déploiement de services depuis le fournisseur jusque chez l'usager doit être garanti. D'autre part, l'exécution locale des services doit être contrôlée, afin de protéger les services de différents fournisseurs les uns des autres.

6. Bibliographie

Apache Software Foundation, "Felix project", http://incubator.apache.org/felix/, 2005.

Czajkowski, G., Daynès, L., Titzer, B., "A Multi-User Virtual Machine", Usenix, 2003.

Fleury, E., Frénot, S., "Building a JMX management interface inside OSGi", Rapport de recherche INRIA n° 5025, 2003.

IBM, "Service Management Framework", http://www.ibm.com/software/wireless/smf/, 2003.

OSGi Alliance, "OSGi Service Platform, Release 4", http://osgi.org, Octobre 2005.

Projet MUSE, IST 507295 FP6, http://www.ist-muse.org, 2004.

Yamasaki, I., "Increasing Robustness by Code Instrumenting: Monitoring and Managing Computer Resource Usage on OSGi Frameworks", OSGi World Congress, 2005.

La négociation du niveau de service dans une architecture de gestion autonome

Nader Mbarek

Université de Bordeaux 1
351, cours de la Libération
F-33405 Talence cedex
Nader.Mbarek@labri.fr

Francine Krief

Université de bordeaux 1
351, cours de la Libération
F-33405 Talence cedex
Francine.Krief@labri.fr

RÉSUMÉ. *Nous proposons un framework pour la négociation du niveau de service dans la gestion des systèmes autonomes. La procédure de négociation se déroule entre des gestionnaires autonomes de haut niveau afin de garantir un niveau de service de bout en bout pour un trafic donné. Dans le framework proposé, une nouvelle opportunité s'offre aux systèmes autonomes grâce à la possibilité de négociation des gestionnaires autonomes de haut niveau avec leurs homologues. Dans un souci de conformité avec les concepts de la gestion autonome, le protocole de négociation SLNP, que nous spécifions, est utilisée dans un environnement standardisé de technologies Services Web*

ABSTRACT. *We propose a framework for service level negotiation in autonomous systems management. The negotiation process occurs between high-level autonomic managers to guarantee an end-to-end service level for specific application traffic flows. In the proposed framework, we provide autonomous systems with a new interaction opportunity thanks to the negotiation with their peers. To be in conformance with the concepts of self-aware management systems, the proposed negotiation protocol called SLNP is used in a Web services environment.*

MOTS-CLÉS : *Négociation, Niveau de service, Gestion autonome, Services Web.*

KEYWORDS: *Negotiation, Service level, Self-management, Web Services.*

1. Introduction

En ce début de 21ème siècle, l'évolution des technologies de la communication a été telle que les réseaux sont présents à tous les niveaux de la société : entreprises, lieux d'habitation mais également individu qui porte sur lui des appareils de communications de plus en plus variés. Parallèlement à cette explosion quantitative des réseaux de communications, il y a une complexification des technologies. Cette croissance du nombre de réseaux et de leurs complexités s'accompagne de difficultés et de problèmes en termes de gestion, de contrôle et de coût de maintenance. Ainsi le coût total de possession des équipements (*TCO : Total Cost of Ownership*) est devenu de plus en plus important (Yankee, 2002). Les techniques de gestion actuelle se basant sur une gestion individuelle et sur des aspects pré-établis se trouvent mises en défaut. En effet, plus le nombre de systèmes interconnectés est important, plus il est difficile d'anticiper les interactions entre leurs composants. Par ailleurs, il devient difficile de gérer rapidement et efficacement l'hétérogénéité de ces composants (Brown et al, 2001).

Un des enjeux des années à venir est donc la mise en place d'un nouveau paradigme qui permettra d'apporter l'autonomie dans les réseaux grâce à un nouveau concept de gestion appelé gestion autonome (*Self Management*). C'est dans ce contexte que nous proposons un *framework* dans lequel nous définissons un protocole de négociation de service et une utilisation des technologies de Services Web pour faciliter les interactions des systèmes autonomes.

Cet article est composé en quatre sections. La première décrit brièvement les concepts de l'autonomie dans les réseaux. Dans la deuxième section, nous présentons l'importance de l'utilisation de la négociation ainsi que des technologies de Services Web pour l'interaction des systèmes autonomes. Dans la section suivante, nous proposons un *framework* pour la négociation du niveau de service dans la gestion des systèmes autonomes. La dernière section présente une plateforme de démonstration pour notre protocole de négociation de niveau de service.

2. Autonomie des réseaux

2.1. *Besoin d'autonomie*

Traditionnellement, la gestion des systèmes et des réseaux est un processus contrôlé manuellement. Il est ainsi nécessaire d'avoir une intervention humaine de la part d'un ou plusieurs opérateurs afin de gérer tous les aspects en relation avec l'évolution dynamique d'un système ou d'un réseau. L'opérateur est alors fortement impliqué dans ce processus de gestion avec des tâches allant de la spécification des politiques de haut niveau à l'exécution de commandes de bas niveau pour la résolution immédiate des problèmes. Bien que cette forme de gestion, où l'homme est considéré comme une composante essentielle de la boucle de contrôle, fût

appropriée dans le passé, il est devenu de plus en plus inadéquat d'intégrer l'intervention humaine dans la boucle de gestion des réseaux. En effet, plusieurs tendances concernant le développement des infrastructures de la technologie de l'information (*IT : Information Technology*) ont compliqué davantage la tâche de gestion des réseaux actuels.

Tous les indicateurs montrent que les infrastructures des technologies de l'information vont évoluer vers une complexité plus importante dans le futur, et la façon la plus adéquate de gérer la complexité de ces systèmes est devenue un sujet primordial. Pour répondre à ce besoin, un consensus semble se dégager et faire converger diverses positions vers des concepts de gestion autonome (*self management)* avec des technologies qui permettent aux systèmes de s'autogérer pour devenir autonome. En effet, il est devenu peu réaliste pour un opérateur humain de maintenir d'une façon fiable le contrôle d'un réseau composé d'un grand nombre de ressources (ordinateurs, serveurs, bases de données, applications, clients...). Il devient alors nécessaire de redéfinir le rôle de l'opérateur humain dans la boucle de gestion. Ainsi, ce dernier ne sera plus impliqué d'une façon directe et interactive dans la prise de décision dans les processus de contrôle et de gestion et va se contenter d'établir les objectifs globaux ainsi que les politiques et directives de haut niveau que doivent respecter les entités mises en jeu dans la nouvelle boucle de gestion et de contrôle.

2.2. *La gestion autonome*

La vision de l'autonomie dans les réseaux est la création d'un système qui sait s'autogérer (*self management system*) sans intervention humaine afin de remédier à la complexité croissante et aux coûts excessifs de la gestion actuelle des réseaux tout en préparant le terrain pour satisfaire les besoins de l'informatique omniprésente du futur. L'idée est aussi de développer des réseaux qui soient autonomes, capables de s'auto organiser à l'instar des systèmes biologiques. Les réseaux deviennent ainsi un ensemble d'entités autogouvernées qui n'ont pas besoin de l'intervention humaine, sauf pour la spécification des directives de haut niveau et des objectifs ce qui permet de cacher à l'administrateur les détails de gestion et de contrôle des composants logiciels et matériels du système autonome. Pour se faire, un nouveau paradigme s'est largement inspiré de ce qui existait dans la biologie et notamment le système nerveux autonome auquel nous devons en partie l'appellation « *Autonomic Computing* » (Horn, 2001) de ce nouveau paradigme. Tout comme l'Homme avec son système nerveux autonome, le réseau avec ses entités autonomes doit être fiable et offrir des garanties de disponibilité, de sûreté, de survie, de sécurité et de maintenance. Ceci peut faire de l'autonomie dans les réseaux un moyen de réaliser l'un des objectifs les plus importants concernant la technologie de l'information à savoir : construire un système sur lequel nous pouvons compter (Sterrit et al, 2003). Ceci peut être rendu possible grâce à une approche holistique ou encore globaliste de ce nouveau paradigme puisqu'il vise à rassembler et harmoniser tous les domaines

de recherche qui peuvent contribuer à la réalisation des réseaux autonomes. Le chemin vers l'autonomie risque d'être long et on prévoit de passer par plusieurs étapes avant d'arriver à l'autonomie totale des réseaux, prévue pour 2020 (FET, 2004), ce qui fait de ce nouveau paradigme une évolution progressive des réseaux de nos jours et non une révolution qui va rompre d'une façon brusque avec les caractéristiques des technologies actuelles.

Ainsi, la manière de concevoir les systèmes autonomes va engendrer une évolution architecturale et fonctionnelle. Les systèmes rempliront des fonctions de gestion autonome (*Self Management*). La liste des objectifs de la gestion autonome a été étendue depuis 2001 (Date de lancement de l'*Autonomic Computing*), néanmoins, le but général reste toujours l'auto configuration (*Self Configuring*), l'auto optimisation (*Self Optimizing*), l'auto restauration (*Self Healing*) et enfin l'auto protection (*Self Protecting*) (Ganek et al, 2003).

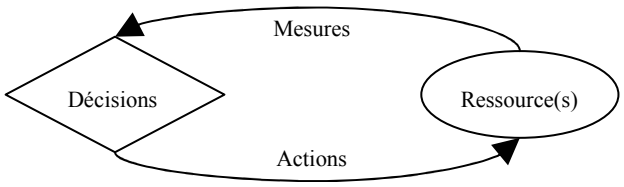


Figure 1. *Boucle de contrôle de la gestion autonome*

Afin de réaliser les objectifs que nous venons de citer, les systèmes autonomes disposent d'une connaissance détaillée de leurs états internes mais aussi de leur environnement (Bustard et al, 2003) et ce grâce à la supervision continue des changements qui peuvent affecter leur environnement. A la détection d'un changement, le système se réajuste d'une façon autonome et la supervision continue afin de déterminer si les nouvelles mesures correspondent aux performances désirées. Cette boucle fermée de contrôle représentée par la figure 1, permet aux entités autonomes de prendre les décisions adéquates et de s'adapter sans l'intervention de l'opérateur et ce grâce aux mesures remontées par les ressources du système autonome ce qui permet ainsi de fermer la boucle. Cette boucle de contrôle est implémentée par des gestionnaires autonomes qui contrôlent des ressources gérées grâce à des interfaces de gestion de type capteur (*sensor*) et effecteur (*effector*) (IBM, 2005).

3. Interactions des systèmes autonomes

Une architecture de gestion autonome est caractérisée par une infrastructure distribuée où toutes les entités autonomes collaborent pour le bon fonctionnement

global du système et où, par la suite, diverses interactions (Kephart et al, 2003) peuvent découler de cette architecture distribuée afin de permettre aux systèmes autonomes de délivrer et de consommer des services avec une garantie d'un certain niveau de service. Ainsi, l'infrastructure distribuée de la gestion autonome doit être orientée services pour supporter les interactions entre entités autonomes afin de représenter aussi bien les services offerts que ceux utilisés par une entité autonome.

3.1. Utilité de la négociation

Les interactions dans un environnement de gestion autonome sont dynamiques et changent avec l'évolution des entités autonomes en termes d'offres et de besoins. De plus, ces interactions peuvent être négociées pour que deux entités ou plus arrivent, par exemple, à un accord commun sur le niveau de service. Cet accord pourra être renégocié si une entité autonome subit un changement qui nécessite une adaptation des accords passés avec ses homologues tout en restant conforme avec les politiques de haut niveau qui la gouvernent.

La relation entre les différents éléments autonomes a été décrite dans (Kephart et al, 2003) comme étant un défi à relever pour les scientifiques qui cherchent à spécifier des systèmes totalement autonomes dans un environnement totalement hétérogène. Il est à noter que les efforts pour standardiser la représentation des accords sur le niveau de service sont en train de se concrétiser avec les travaux portant sur le SLA, SLS (Goderis et al, 2003), WSLA et bien d'autre. Cependant un manque important se fait sentir au niveau de la spécification des protocoles pour établir les règles de négociation et gouverner les flux de messages de négociation entre les entités autonomes mises en jeu mais aussi au niveau de la spécification des stratégies de négociation. Une négociation peut concerner plusieurs systèmes autonomes, elle est alors multilatérale et peut porter sur plusieurs attributs dont les valeurs doivent être garanties en cas de succès de la négociation. Nous allons revenir plus en détails sur les aspects liés à la négociation, comme une possibilité d'interaction dans les systèmes autonomes, lors de la spécification de notre protocole SLNP dans la section 4.3.

3.2. *Utilisation des Services Web*

La garantie de réussite de l'évolution vers l'autonomie totale passe nécessairement par l'adoption massive de standards ouverts pour l'interaction et la réalisation des différents composants de l'architecture de gestion autonome. En effet, différentes ressources se rapportant aux technologies de l'information sont produites par différents constructeurs avec des technologies loin d'être interopérables en termes de contrôle et de gestion. Ceci a produit une multitude de systèmes de gestion utilisant des technologies différentes dont l'intégration est de plus en plus difficile et de plus en plus coûteuse. La réalisation des différentes fonctions de gestion dans cet

environnement hétérogène comportant des ressources différentes et distribuées est par conséquent devenue particulièrement complexe. Une solution serait l'utilisation des Services Web pour remédier aux problèmes d'intégration et permettre d'homogénéiser des systèmes de gestion qui se basent sur des implémentations et des plates-formes différentes. Il s'agit, par conséquent, d'adopter et d'améliorer les standards ouverts des technologies de Services Web afin de pouvoir les utiliser dans le cadre de la gestion autonome.

Des travaux ont été réalisés dans ce sens au sein de l'organisme de standardisation OASIS (*Organization for the Advancement of Structured Information Standards*) par les différents participants (IBM, HP, CA, Hitachi, Dell, AmberPoint, BEA, BMC...) du comité technique WSDM TC. Ces travaux ont été effectués en collaboration avec d'autres groupes de standardisation au sein même d'OASIS mais également avec le DMTF (*Distributed Management Task Force*), le GGF (*Global Grid Forum*) et surtout avec le W3C (*World Wide Web Consortium*) car plusieurs des technologies de Services Web (WS : *Web Services*) qui seront utilisées dans le cadre de la gestion autonome ont été standardisées par ce consortium. Ces travaux ont abouti à la définition du standard WSDM (*Web Services Distributed Management*). Celui-ci comporte deux spécifications nommées MUWS (*Management Using Web Services*) (Vambenepe, 2005) et MOWS (*Management Of Web Services*) (Sedukhin, 2005) afin de couvrir les deux domaines se rapportant à la gestion et aux Services Web, c'est-à-dire la gestion en utilisant les Services Web et la gestion des Services Web eux-mêmes.

La première spécification de WSDM, appelée MUWS, peut servir de base commune pour l'implémentation des interfaces de gestion de type capteur ou encore effecteur et va permettre le contrôle des ressources gérées par les gestionnaires autonomes en utilisant les technologies de Services Web. L'adoption de cette spécification permet l'amélioration de l'interopérabilité des différentes solutions de gestion autonome qui vont supporter les mêmes technologies standardisées de Services Web pour le contrôle des ressources gérées. MUWS fournit un cadre commun et flexible permettant de spécifier des possibilités de gestion qui seront exposées par les ressources gérées. Ces possibilités de gestion sont extensibles, ce qui permet la réalisation des différents objectifs de la gestion autonome.

WSDM fait appel à de nombreuses spécifications WS émanant du W3C et d'OASIS telles que *Web Services Architecture*, XML, XML Schema, WSDL (*Web Services Description Language*), SOAP (*Simple Object Access Protocol*), *WS Base Notification* et *WS Resource Properties*. Ainsi pour définir des possibilités de gestion accessibles via les technologies WS, le standard WSDM s'appuie, entre autres, sur les spécifications que nous venons de citer pour l'envoi de messages, la description, la découverte, les notifications et l'accès aux propriétés des ressources. Ce standard est d'un grand intérêt dans le cadre de la gestion autonome puisque son utilisation permet d'uniformiser les interactions entre gestionnaires autonomes et ressources gérées grâce à des technologies de Services Web interopérables.

4. Cadre de travail : proposition d'un Framework

Dans cette section, nous exposons un cadre de travail pour faciliter les interactions au sein même d'un système autonome (1) ou encore entre plusieurs systèmes autonomes différents (2). Pour se faire, nous spécifions un protocole de négociation et nous utilisons les technologies de Services Web pour rendre possible deux types d'interactions, à savoir les interactions horizontales (2) et les interactions verticales (1). La figure 1 représente le *framework* que nous proposons. La définition des différentes entités impliquées dans les divers processus d'interaction est donnée dans la section suivante.

4.1. *Définition des composants*

Le *framework* que nous proposons contient les éléments suivants :

- *HAM* : *High-level Autonomic Manager*; le gestionnaire autonome de haut niveau utilise un protocole de négociation pour communiquer avec d'autres HAM afin d'obtenir un accord portant sur un niveau de service particulier. De plus, il contrôle un ou plusieurs LAM en utilisant une interface de gestion standardisée basée sur les technologies de Services Web.

- *LAM* : Low-level Autonomic Manager; le gestionnaire autonome de bas niveau contrôle une ou plusieurs ressources gérées grâce au même type d'interface utilisé par le HAM pour le gérer.

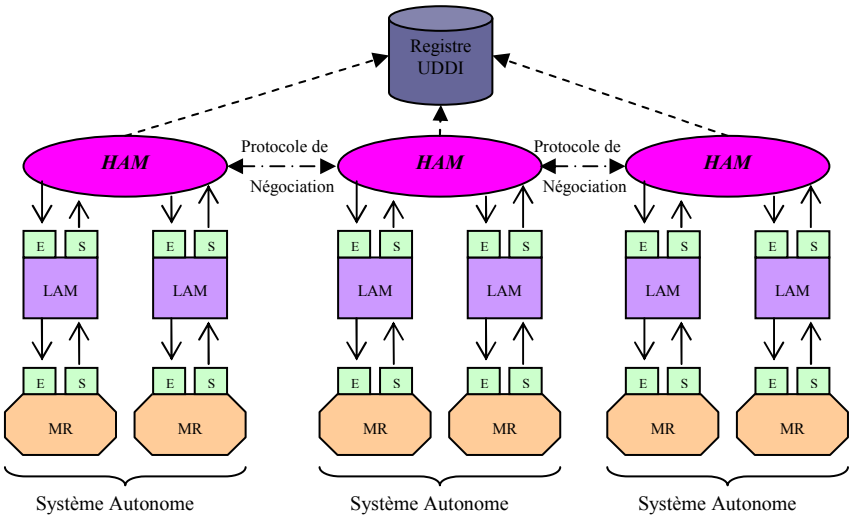


Figure 2. *Interactions des systèmes autonomes*

- *MR* : *Managed Resource(s)*; la ressource gérée peut être une entité logicielle ou matérielle (serveur, routeur, base de données...). Une ou plusieurs ressources gérées exposent leurs possibilités de gestion par le biais d'une interface de gestion composées des deux interfaces capteur (Sensor) et effecteur (Effector).

- *S* : Sensor; l'interface de type capteur permet au HAM et au LAM de demander ou recevoir des notifications d'informations de monitorage provenant, respectivement, des LAM et des MR dont ils sont responsables.

- *E* : Effector; L'interface de type effecteur donne au HAM et au LAM les moyens d'accomplir des actions afin de modifier le comportement des entités qu'ils gèrent.

- *Registry* : c'est un registre de type UDDI (Bellwood et al, 2002). Les gestionnaires autonomes de haut niveau (HAM) utilisent ce registre afin de publier les adresses des Services Web dont ils disposent et faciliter ainsi leurs découvertes et leurs interactions.

- *AS* : Autonomous System; un système autonome est un ensemble de gestionnaires autonomes, de ressources gérées ainsi que d'interfaces de gestion. La nature du système autonome dépend de la nature des ressources gérées qu'il contient. Ceci nous permet d'utiliser ce composant pour représenter différents types de systèmes allant d'un Elément Autonome à un Domaine Autonome. Un système autonome est un système capable de s'autogérer.

4.2. Concepts du Framework

L'objectif de ce travail est la spécification des concepts d'interactions entre systèmes autonomes afin de garantir un niveau de service pour un trafic généré entre un AS d'entrée et un AS de sortie. Pour atteindre ce but, nous spécifions dans notre *framework* deux types d'interactions comme le montre la figure 2. Le premier type d'interaction est une interaction horizontale inter systèmes autonomes. Dans ce type d'interaction, un AS d'entrée initie par l'intermédiaire de son gestionnaire autonome de haut niveau (HAM) un processus de négociation pair à pair avec les HAM adjacents situés sur le chemin du HAM de sortie. La communication entre les HAM est basée sur un protocole de négociation que nous spécifions dans la section 4.3. Ce protocole de négociation utilise les technologies de Services Web. Les HAM qui offrent la possibilité de négocier un niveau de service, doivent préalablement publier leurs Services Web afin de rendre possible cette négociation. Suite au succès de la procédure de négociation, chaque gestionnaire autonome de haut niveau (HAM) aura la responsabilité de garantir le niveau de service correspondant dans le système autonome auquel il appartient. Cette garantie est rendu possible grâce au deuxième type d'interaction : il s'agit d'une interaction verticale intra systèmes autonomes. Dans ce type d'interaction, un gestionnaire autonome de haut niveau (HAM) fournit

aux gestionnaires autonomes de bas niveau (LAM), dont il est responsable, le niveau de service négocié et accepté afin qu'ils modifient la configuration de leurs ressources gérées conformément aux attributs du niveau de service qui doit être garanti pour le trafic qui va utiliser les ressources du système autonome. L'interface de gestion qui permet ce type d'interaction entre les HAM et les LAM mais aussi les LAM et les ressources gérées est basée sur le standard WSDM. La spécification MUWS du WSDM permet une implémentation standardisée des interfaces capteur et effecteur tout en assurant une meilleure interopérabilité entre différentes technologies hétérogènes.

4.3. Spécification du protocole de négociation SLNP

Les systèmes autonomes qui ont la capacité de s'autogérer ont besoin de coopérer avec leurs homologues afin d'offrir et utiliser des services. Une possibilité pour arriver à un accord sur le niveau du service qu'ils veulent utiliser ou offrir est l'utilisation d'un protocole de négociation. Cette possibilité est considérée comme un défi de recherche (FET, 2004) (Kehpart, 2003) pour le nouveau paradigme de gestion autonome. Nous proposons dans ce sens, un protocole de négociation que nous appelons SLNP (*Service Level Negotiation Protocol*) adapté à un environnement de gestion autonome. Le protocole SLNP utilise les technologies de Services Web afin de permettre une négociation multilatérale entre les gestionnaires autonomes de haut niveau appartenant à des systèmes autonomes différents.

Les besoins que nous définissons pour notre protocole SLNP découlent des spécificités d'une procédure de négociation. Ainsi, il est important de permettre aux entités de négociation, c'est à dire les HAM, d'initier, modifier ou encore résilier un niveau de service déjà accepté à la suite d'une négociation antérieure. De plus, notre protocole doit être indépendant des paramètres de niveau de service afin de faciliter son extension par la définition de nouveaux paramètres de QoS, mobilité et sécurité (Benmammar, 2003) sachant que nous définissons un schéma XML (Thompson, 2001) pour le niveau de service afin d'assurer une compréhension globale de celui-ci dans les différents HAM mis en jeu dans une procédure de négociation.

La terminologie adoptée pour l'utilisation de SLNP dans le *framework* proposé est la suivante :

-SHE (*SLNP HAM Entity*): une entité de type HAM qui supporte SLNP.

-SHI (*SLNP HAM Initiator*): une SHE qui initie le processus de négociation.

-SHR (*SLNP HAM Responder):* la dernière SHE sur le chemin de la négociation.

-SHF (*SLNP HAM Forwarder*): une SHE, située entre une SHI et une SHR, qui participe au processus de négociation.

Une ou plusieurs SHF peuvent être présentes sur le chemin de la négociation. Nous attirons l'attention sur le fait qu'une SHE peut être considérée comme une

SHI, SHF ou SHF suivant le rôle qu'elle joue dans un scénario de négociation. Les messages SLNP sont envoyés, de proche en proche, aux différentes SHE présentes sur le chemin vers la SHR et participant à la procédure de négociation et ce en utilisant les technologies de Services Web. Ainsi, lorsqu'une SHE reçoit un message SLNP, par l'intermédiaire de son interface de Service Web de négociation, un nouveau message (différent ou non) est généré et le Service Web de l'entité SHE adjacente est invoqué pour lui transmettre le message en question. Nous utilisons WSDL (Web Services Description Language) (Christensen, 2001) pour la description du Service Web de négociation d'une SHE et SOAP (Simple Object Access Protocol) (Box, 2000) pour l'invocation du Service Web en question.

4.3.1 Modèle de fonctionnement de SLNP dans un framework de gestion autonome

A la réception d'une requête de négociation par une SHE, une première interaction verticale a lieu avec les gestionnaires autonomes de bas niveau (les LAM) correspondants afin de vérifier la disponibilité des ressources pour le niveau de service demandé. La décision prise par une SHE pour répondre au message SLNP (accepter, refuser, valeurs alternatives...), ou encore la modification ou non du message qu'elle va transférer, dépend de cette interaction avec les LAM mais aussi des politiques de haut niveau, directives et autres types d'informations contenues dans le plan de connaissance du HAM correspondant. Lorsque la procédure de négociation aboutit à un accord, le niveau de service est identifié et une deuxième interaction verticale entre les différents SHE et leurs LAMs permettra de faire passer à ces derniers les valeurs des paramètres de niveau de service à garantir. En se conformant à ces valeurs négociées par les HAM avec ses homologues, les LAMs vont réaliser une fonction d'auto configuration grâce à une autre interaction verticale de bas niveau avec leurs ressources gérées en utilisant leurs interfaces de gestion. Chaque gestionnaire autonome de bas niveau (LAM) va essayer de garantir la partie locale du niveau de service négocié de bout en bout. Cette partie correspond aux capacités disponibles de ses ressources gérées qui ont influé la prise de décision du SHE correspondant suite à la première interaction verticale. L'interface capteur des ressources gérées fournit aux LAMs les mesures relatives aux flux dont le niveau de service doit être garanti. Une fonction d'auto optimisation peut avoir lieu pour améliorer l'utilisation des ressources dans l'offre de services avec un niveau négocié et accepté. Lorsqu'une violation du niveau de service est détectée, une fonction d'auto restauration est déclenchée et des actions correctives sont exécutées pour résoudre ce problème. Si ce problème n'est pas résolu, l'interface capteur du LAM va notifier le HAM correspondant. Ce dernier, ayant une vue de plus haut niveau que ses LAM, va entreprendre d'autres actions correctives en utilisant l'interface de type effecteur des LAM qu'il contrôle. Si le problème est persistant, ce dernier va notifier les autres SHE concernées par la dégradation éventuelle du niveau de service et un nouveau processus de négociation peut être initié afin de modifier ou encore résilier le niveau de service en question. Il est à noter que les fonctions de gestion autonome, qui peuvent avoir lieu pour garantir les engagements pris par les SHE suite à la procédure de négociation avec notre protocole SLNP, peuvent être

réalisées par les gestionnaires autonomes de bas niveau qui ont une vision restreinte, mais aussi par le gestionnaire autonome de haut qui a une vision plus globale et qui peut collaborer avec ses homologues des autres systèmes autonomes afin de réaliser les différents objectifs d'autonomie.

4.3.2 Messages SLNP

Nous définissons pour le protocole SLNP six types de messages permettant de satisfaire les besoins d'une procédure de négociation :

- ***Negotiate*** : c'est un message généré par une SHI à destination d'une SHR. Il permet de spécifier les valeurs et les paramètres de niveau de service qui vont être négociés. Ce message est interprété par l'entité SHF adjacente sur le chemin de la négociation et un autre message *Negotiate* (modifié ou non) est envoyé vers le Service Web de la prochaine SHF jusqu'à l'arrivée à la SHR.

- ***Revision*** : c'est un message envoyé par l'entité SHR vers la SHI afin de proposer une alternative aux paramètres et/ou valeurs reçues dans un message *Negotiate*.

- ***Response*** : c'est un message généré par l'entité SHR ou SHI suite à la réception d'un message contenant un élément *Response Request* afin d'accepter ou refuser une demande de négociation ou de révision ce qui permet de terminer le processus de négociation dans des délais raisonnables. Un message *Response* positif doit obligatoirement contenir un identificateur (ID) du niveau de service accepté.

- ***Modify*** : c'est un message généré par une SHI à destination d'une SHR avec l'identificateur du niveau de service enregistré après une procédure de négociation réussie. Cette demande de modification peut être négociée ou alors une réponse immédiate peut être demandée par la SHI en utilisant l'élément *Response Request* et le processus de négociation se termine dans ce cas par un message *Response* venant de la SHR.

- ***Notify*** : c'est un message envoyé par une SHE autre que la SHI pour notifier la SHI d'une dégradation du niveau de service qu'elle garantissait localement. Ce message contient obligatoirement un élément *Response Request*. Ce message peut être considéré comme un message particulier puisque la SHE qui l'utilise invoque directement le Service Web de la SHI sans passer par les Services Web des autres SHE. Cette notification de dégradation peut être accompagnée par des pénalités mais aussi par la demande de modification de la part de la SHI pour vérifier si les autres SHE peuvent compenser cette dégradation.

- ***Release*** : c'est un message généré par une SHI à destination d'une SHR pour résilier un niveau de service déjà négocié dans une procédure de négociation antérieure. Il contient obligatoirement l'identificateur du niveau de service en question. Les informations correspondantes à ce niveau de service seront alors détruites du plan de connaissance de chaque SHE dès réception d'une réponse positive.

Le diagramme d'état ou encore *Finite State Machine* (FSM) que nous définissons pour une entité SHE comporte trois états : E0 est l'état libre dans lequel le gestionnaire autonome de haut niveau n'a pas encore commencé la procédure de négociation en utilisant SLNP; E1 est l'état pendant lequel une SHE est en attente d'un message SLNP d'une autre entité SHE et enfin l'état E2 représente la fin de la négociation. La transition d'un état à l'autre est le résultat de la réception et/ou l'envoi d'un message SLNP. Nous ne nous n'intéressons dans la spécification du FSM d'une SHE ni aux interactions externes de cette dernière avec le gestionnaire autonome de bas niveau ni aux interactions internes telles que les *timers*.

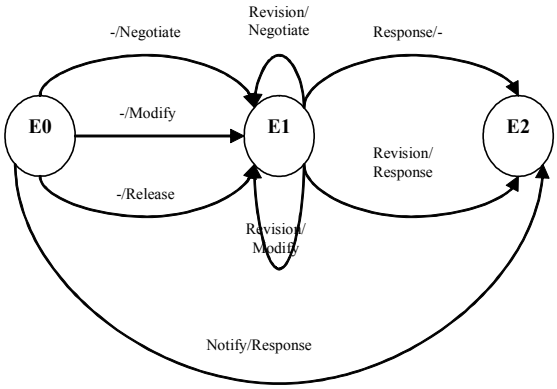


Figure 3. *FSM d'une entité SHI*

La figure 3 représente le diagramme d'état d'une SHI. Trois messages, *Negotiate, Modify* et *Release* font passer une SHI de l'état E0 à l'état E1 où elle est en attente d'un message *Revision* ou *Response*. Seule la réception d'un message *Response* fait passer la SHI à l'état E2 et la négociation est terminée, alors que la réception d'un message *Revision* ne change pas l'état de SHI et la négociation est poursuivie. Finalement dans le cas particulier de réception d'un message *Notify* lorsque la SHI est dans l'état E0, cette dernière passe à l'état E2 suite à l'envoie obligatoire d'un message *Response* vue la présence de l'élément *Response Request*.

5. Plateforme de démonstration

Nous sommes en train de concrétiser par une plateforme de démonstration le *framework* que nous proposons pour la négociation du niveau de service entre systèmes autonomes avec SLNP et ce grâce à notre participation dans l'élaboration de l'architecture SWAN que nous décrivons dans ce qui suit.

5.1. L'architecture SWAN

Self-aWare mAnageNt (SWAN) (Swan, 2006) est un projet exploratoire de recherche qui a pour objectif la définition de nouvelles méthodes pour la gestion autonome. Afin de réaliser cet objectif, nous élaborons une architecture de gestion autonome multi domaines (figure 4) pour la provision et le monitorage de services avec garantie de QoS. Dans cette architecture le niveau de service va être négocié pour le trafic généré par une application de Visio Conférence d'un utilisateur final et traversant trois domaines hétérogènes et autonomes

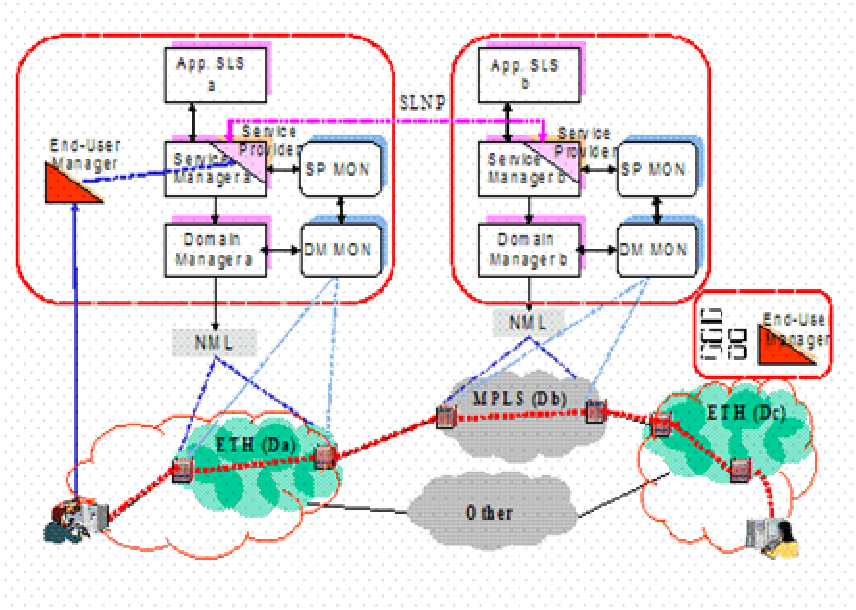


Figure 4. *L'architecture SWAN*

Nous décrivons dans ce qui suit les composants de cette architecture qui implémente la vision du *framework* que nous proposons avec notre protocole de négociation SLNP. Dans l'architecture SWAN, l'entité AS que nous définissons dans le *framework* n'est autre qu'un domaine autonome avec un ensemble de gestionnaires et ressources du réseau correspondants. Nous distinguons deux types de gestionnaires dans cette architecture : Le *Domain Manager* (DM) et le *Service Manager* (SM). Le DM assure la fonction de gestionnaire autonome de bas niveau (LAM) décrite dans le *framework*. Le SM ou encore le *Service Provider* (SP) représente un même composant. Ce composant est appelé SM ou SP suivant qu'il assure respectivement, la fonction d'interaction verticale du HAM décrite dans le *framework* et qui se traduit par le contrôle du DM dans le cas de l'architecture SWAN, ou alors la fonction d'interaction horizontale du HAM qui se traduit par la

procédure de négociation du niveau de service avec ses homologues SP des autres domaines autonomes de l'architecture SWAN. L'entité de monitorage (DM-Mon) est l'implémentation de l'interface capteur que le DM utilise pour retrouver les mesures concernant les ressources réseaux dont il est responsable. Nous utilisons les Services Web pour toutes les interactions entre ces différentes entités. Les Services Web que nous utilisons pour l'implémentation des interfaces de gestion ne sont pas encore conformes au standard WSDM tel que nous l'avons défini dans le *framework* proposé. L'adoption de ce standard dans le futur rendra les composants de notre architecture encore plus interopérables.

5.2. Utilisation de SLNP dans l'architecture SWAN

Dans le cadre de l'architecture SWAN, le protocole SLNP est utilisé pour la négociation du niveau de service entre les trois SP présents dans cette architecture. Le niveau de service que nous voulons garantir pour une application de Visio Conférence est représenté par un SLS (*Service Level Specification*) qui contient un ensemble de paramètres que les SP vont négocier en utilisant SLNP.

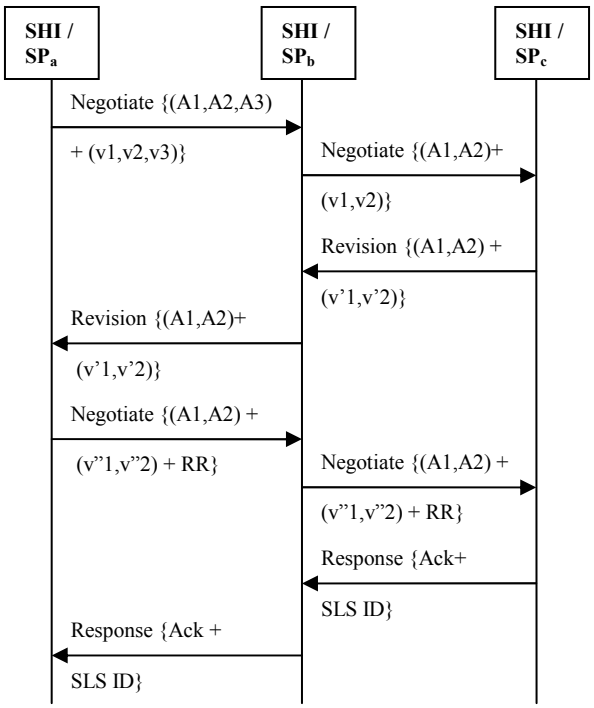


Figure 5. *Exemple d'un MSC de SLNP*

Parmi ces paramètres, certains ont été définis dans un modèle de SLS (Goderis, 2003) élaboré par le projet Tequila, d'autres ont été utilisés par des protocoles de négociation tels que COPS SLS (Thi, 2002) et DSNP (Chen, 2002). Contrairement à ces protocoles de négociation, SLNP est basée sur les technologies de Services Web qui lui garantissent une interopérabilité nécessaire lorsque nous voulons utiliser un protocole de négociation dans un environnement de gestion autonome avec des systèmes autonomes utilisant des plateformes et technologies hétérogènes.

Les paramètres SLS transportés par SLNP sont conformes à un schéma XML que nous avons spécifié afin de garantir la compréhension du niveau de service par les différents SP. Les paramètres les plus importants que nous utilisons sont le temps de service, l'identification du trafic, le scope, la garantie de performance (bande passante, délai, gigue, taux de perte), la description et la conformité du trafic (*Token Bucket*), le traitement d'excès, le mode de négociation (SLS predefini ou non), la fiabilité (MDT : *Mean Down Time* ; MTTR : *Mean Time To Repair*), l'intervalle de renégociation.

La figure 5 représente par un MSC (*Message Sequence Chart*) un scénario simple de négociation avec SLNP afin de faciliter la compréhension du fonctionnement de ce protocole. L'entité SHI qui n'est autre que le SP_a initie la négociation avec un message *Negotiate* contenant les attributs SLS (A1, A2, A3) avec les valeurs (V1, V2, V3). Après la réception de valeurs alternatives proposées par l'entité SHR qui correspond au SP_c, la négociation se termine avec l'utilisation de l'élément *Response Request* qui entraîne l'envoie d'un message *Response* avec un identificateur du niveau de service (SLS ID) accepté par les différents SPs. Dans le scénario présenté dans la figure 5, la négociation a abouti à un niveau de service caractérisé par les attributs (A1, A2) avec les valeurs alternatives (V"1, V"2).

6. Conclusion et perspectives

Dan cet article, nous avons présenté un *framework* qui montre l'importance de l'utilisation d'un protocole de négociation et des technologies standardisées de Services Web afin de permettre les différentes interactions horizontales et verticales des systèmes autonomes. La négociation du niveau de service que notre protocole SLNP assure d'une façon interopérable en se basant sur les standards des technologies de Services Web facilite la collaboration entre différents systèmes de gestion autonome, déployés sur des plateformes hétérogènes et utilisant des technologies différentes, pour l'offre et l'utilisation des services avec des garanties en terme de QoS.

Après avoir spécifié le protocole SLNP, défini par un schéma XML les paramètres SLS qu'il va utiliser et décrit en utilisant WSDL les interfaces de Services Web de négociation qui vont être invoquées par les messages de SLNP grâce à SOAP, nous allons maintenant essayer d'intégrer notre protocole de négociation dans l'architecture SWAN. Nous allons alors nous intéresser

particulièrement au temps pris par une procédure de négociation pour aboutir à un accord ou à un échec tout en essayant d'évaluer l'impact de l'utilisation des Services Web dans ce temps moyen de négociation.

7. Bibliographie

Bellwood T., Clément L., Ehnebuske D., Hately A. "Universal description, discovery and integration (uddi) specification". Technical report, OASIS Committee, Juillet 2002.

Benmammar B., Thi Mai Trang N., Pujolle G., Yilmaz V. "Définition d'un SLA / SLS", Document IP-SIG, Decembre 2003.

Box D., et al., « Simple Object Access Protocol (SOAP) 1.1", W3C Note, mai 2000.

Brown A., Patterson D., "To Err Is Human", *EASY'01*, Suède, Juillet 2001.

Bustard DW., Sterritt R., "Towards an autonomic computing environment", *Proceedings IEEE DEXA 2003 workshops*, Prague, p. 694-698, septembre 2003.

Chen J., et al, "Dynamic Service Negotiation Protocol", IETF draft, Decembre 2002.

Christensen E., et al, « Web services Description Language 1.1", W3C Note, mars 2001

FET, "New communication paradigms for 2020",meeting report, Bruxelles, Mars 2004.

Ganek A.G., Corbi T.A, "The dawning of the autonomic computing era", *IBM Systems Journal*, vol. 42, No 1, 2003, p. 5-18.

Goderis D., Griffin D., Jacquenet C., Pavlou G. "Attributes of a service level specification template", IETF, draft-tequila-sls-03, Octobre 2003.

Horn P., "Autonomic computing: IBM perspective on the state of information technology", IBM T.J.Watson Labs, NY, Présenté dans AGENDA'01, Scottsdale, Octobre 2001.

IBM Group, "An architectural blueprint for autonomic computing", White paper, Juin 2005.

Kephart J.O., Chess D.M., "The vision of autonomic computing", *IEEE Computer Society*, vol. 36, No 1, Janvier 2003, p. 41-50.

Sedukhin I., "Web Services Distributed Management: Management of Web Services (MOWS) 1.0", OASIS Standard, mars 2005.

Self-aWare mAnageNt (Swan) RNRT project accessible at: http://swan.elibel.tm.fr/

Sterritt R., Bustard D., "Autonomic computing : a means of achieving dependability?", *IEEE ECBS'03*, Huntsville, p. 247-251, Avril 2003, p. 247-251.

Thi Mai Trang N., Boukhatem N., Ghamri Y., Pujolle G., "Service Level Negotiation and COPS-SLS Protocol", *IEEE Communication Magazine,* May 2002, pp. 158-165.

Thompson H.S., et al., "XML Schema Part 1: Structures", W3C Recommandation, mai 2001.

Vambenepe W., "Web Services Distributed Management: Management using Web Services (MUWS 1.0) Part 1", OASIS Standard, Mars 2005.

An Informational Framework for Autonomic Networking

Z. B. Daho, N. Simoni, C. Yin, F. Bennani
GET / ENST / INFRES / CNRS / LTCI / UMR 5141
46, rue Barrault
75013 Paris – France
{benahmed, simoni, yin, fbennani}@enst.fr

ABSTRACT. *The emerging next generation networks and services exhibit high performance and novel functionalities. The efficient management of this context introduces new challenges: fully operational management automation, operational flexibility enhancement, related costs mastery and management integration to achieve an end-to-end solution. A truly distributive management system, up to the point that each function is self-managed, is an attractive candidate solution to address these challenges. To reach this solution, in this paper we present a framework and we show how its models can be used to allow the integration and automation of processes within each service and network component.*

RESUME. *L'émergence des réseaux et services de nouvelle génération a mis en exergue de hautes performances et de nouvelles fonctionnalités. La gestion efficace de ce contexte présente de nouveaux défis : l'automatisation complète de la gestion, l'augmentation de la flexibilité opérationnelle des traitements, la maîtrise des couts et l'intégration de la gestion pour concevoir une solution complète de bout en bout. Un système de gestion totalement distribué, où chaque entité fonctionnelle s'autogère, est une solution intéressante pour relever ces défis. Pour atteindre cette solution, nous présentons dans cet article un cadre informationnel et nous montrons comment ses modèles peuvent être utilisés pour permettre l'intégration et l'automatisation des traitements de tous les composants de service et du réseau.*

KEYWORDS: *Autonomic networking, automation, autonomous entity, meta-model, model, informational framework*

MOTS-CLÉS: *Gestion de réseaux et services autonomes, automatisation, entité autonome, méta-modèle, modèle, cadre informationnel*

1. Introduction

Nowadays, telecom players are massively taking up globalization strategies to face the increasingly competitive context. In a chaotic and unpredictable market, they have to support new network and service challenges while reducing their operational costs in order to maximize their profits. But, unfortunately most of today's network management systems are often centralized where a manager queries agents, builds a view of the managed network and takes decisions if a problem is detected. The drawbacks of this centralized architecture increase as the network grows in size and complexity.

To alleviate the centralized manager and increase management scalability, distributed architectures have been proposed over the last few years using different technologies (Boutabaa *et al.*, 2002) (Martin-Flatin *et al.*, 1999). Recent advances in networking management propose to integrate management intelligence (capabilities) within the managed entities leading to build self-management systems. However, this is not enough to support the complete automation of the service delivery chain transparently; that is, from the customer service request, through service configuration and network provisioning, to invoicing and billing. This complete process automation is desired by the operators to reduce their operational costs and their Time to Market. This is why, we propose the new concept of *autonomic networking* in which the aim is to create autonomous entities that self-manage their own behaviors and collaborate to reach management objectives as well as the end-to-end contracted SLA.

Section 2 identifies requirements for a complete solution for autonomic networking. The related work is presented in section 3 and analyzed, in section 4, according to the identified requirements. As it will be shown in the later section, the existing solutions mainly concern the network level and the automation of fault, configuration, performance and security management functionalities. The related work gives partial answers to our requirements.

The aim of this paper is to propose a model-driven framework for autonomic service and network management. The proposed framework is presented in section 5. The main feature of the proposed solution is that it is based on a meta-model to build the self-managing components to reach the end-to-end automation objective. An example concerning the application of the proposed framework is presented in section 6. Section 7 concludes the paper by summarizing the key points of our proposals and gives the main future orientations.

2. Autonomic networking requirements

An autonomic networking solution aims to create autonomous entities that self-control their own behavior and collaborate to reach management objectives

according to the contracted QoS. This autonomic behavior needs not only the automation of service and network functionalities, but also the automation of the exchanges between the CRM, the Service Management and the Network Management organizations that constitute the BSS and the OSS. To that end, the following requirements have been identified: (*i*) The first requirement concerns the *autonomy* feature of network and service entities. The autonomy feature allows the entity to react automatically according to the detected events as well as to its related states. (*ii*) The second requirement concerns the *automation* feature. Operators have to deliver services rapidly and in an efficient manner. Therefore, they have to support the complete automation of the service delivery chain. (*iii*) The third requirement concerns the *organization* of the distributed management capabilities. We have to find the best manner to distribute the management capabilities as well as behavior information. This distribution has to be done while taking into account organizational constraints. (*iv*) The fourth requirement concerns the communication protocols used by the autonomous entities to fulfill the global objective. (*v*) The last requirement concerns the architectural aspects. We have to find how these autonomous entities, which belong to different management levels, can constitute a coherent structure to meet the aimed behavior and the aimed management objectives.

3. Self-management related work

To face the explosive growth in communication, information, service and network complexities, operators look for new management solutions. Many studies aim to fully automate the management functionalities. As IBM (Kephart *et al.*, 2003), many researches were influenced by biological self-managing systems. Self-management systems aim to automate the whole management chain by providing automatic capabilities to service and network entities. Current proposals mainly aim to handle autonomously fault (self-healing), configuration (self-configuring), performance (self-optimizing) and security (self-securing) management functions (White *et al.*, 2004) (Sterritt *et al.*, 2003). Self-healing concerns the fault management. Self-healing system entities can ensure their own identification, analysis and effective fault resolution, recovery and repair. Self-configuring concerns configuration management and addresses the need to make the system entity configuration or reconfiguration more dynamic and autonomic and thus self-define configuration parameters and perform them. Self-optimization concerns performance management of systems and resource usage optimization. The system seeks to continuously evaluate and optimize its behavior in proactive or reactive ways in order to measure the provided performance and to take dynamic decisions to meet the contracted agreements. Self-secure or self-protection concerns the security capability supported by the systems. Self-securing systems aim to develop self-authentication, self-authorization and self-audit capabilities. Thus, systems can defend themselves from both internal and external malicious attacks as well as unauthorized operations.

4. Analysis of existing proposals

In this section we will analyze the existing proposals according to the identified requirements. (*i*) Concerning the autonomy feature, most of existing solutions react to specific network events (faults, configuration, etc.). However, we need not only the autonomy of the service and its components to fulfill in the most flexible manner the end-to-end service delivery chain and to support the different situations, but also to react not just to specific events but rather to any event that affects the behavior of service and network entities. Therefore we need to consider non-functional information (QoS) of the whole networking entities. As a consequence, each entity will be able to react according to its own (local) objective and contract. (*ii*) Concerning the end-to-end automation, the existing autonomic solutions mainly concern the network level and the automation of fault, configuration, performance and security management functionalities. Therefore, for the end-to-end automation, this is not sufficient. We need service processes and the automation of management functions within the service level and a manner to orchestrate/coordinate the end-to-end tasks to perform the global service delivery chain. (*iii*) Concerning the organizational aspects, we have seen that the actual solution tends to distribute the management intelligence over the agents allowing them to have reaction and management capabilities. However, we still need to organize this distribution because we have to introduce the solution within a specific operator context, so that its constraints and policies have to be taken into account. (*iv*) Concerning the communication aspects, current solution still uses client/server protocols for the communication. Recently, new autonomic solutions use peer-to-peer protocols to make automatic resource discovery and resource sharing easier. The both communication protocols have advantages and disadvantages. A good solution will try to smartly use both client/server and peer-to-peer protocols to optimize the communication. (*v*) Finally, the architecture of the target solution should cover the whole management levels to ensure the end-to-end automation of the service delivery chain.

Hereafter, we respond to these new requirements by proposing a model-driven framework for autonomic service and network management.

5. Towards a model-driven framework for autonomic networking

The proposed framework is based on a meta-model presented in section (§5.1). This meta-model gives the foundation structure of each autonomous element within all management levels (from BML to EML). As meeting the SLA is one of the main operators' objectives, the QoS is taken into account within the proposed meta-model. Starting from the proposed meta-model, five dimensions for autonomic networking are proposed (§5.2). Each dimension drives the automation within a specific management aspect.

5.1. The "Autonomous Component" meta-model: the framework core entity

The structure of each self-managed element is given by the "Autonomous Component" meta-model. The term "Autonomous Component" refers to elements that belong to any management level. Within each level, we differentiate between (*i*) the *nodes*: the elements that fulfill processing functionalities like caching within the service level, routing within the network level and low-level data processing for equipments. (*ii*) the *links*: the elements that fulfill transmission functionalities like virtual links that support the interactions between two service elements, logical links that support the flow between two network end points and physical links that support the communication between two physical equipments. No matter what their management level is, these nodes and links have the same structure given by the "Autonomous Component" meta-model shown in figure 1.

Each component has a base service interface and a management interface. Through the base interface, the autonomous component provides its operational service. Through the management interface, the autonomous component proposes management services like tuning the QoS parameters, setting the management rules and sending notifications. The integration of management makes it possible for a managed object to be configured as a delegated autonomous agent owning the problem-solving capabilities which are required by the autonomic management.

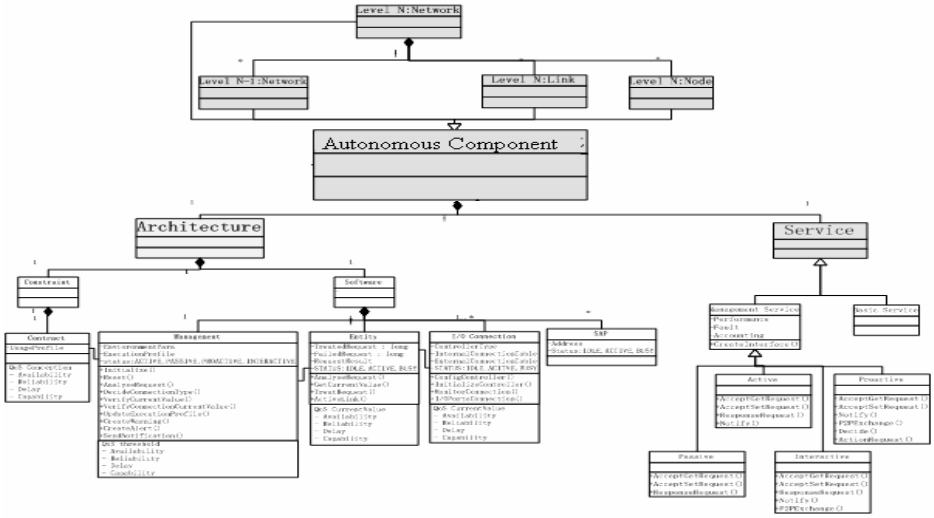


Figure 1: *The structure of an autonomous component: the meta-model*

Concerning the autonomous component behaviors, it is necessary to have a homogeneous expression of their QoS in order to evaluate the end-to-end behavior. By defining and integrating non-functional information (QoS) within the meta-

model, it becomes possible to get an image of each element's behavior and to aggregate the QoS information across all the management levels to have the end-to-end behavior (QoS traceability). The proposed meta-model is the core concept of the framework. Starting from the meta-model and its concepts, five dimensions are analyzed. This is detailed in the following section.

5.2. Five dimensions for autonomic networking

In this section, five dimensions are described for autonomic networking. The architectural dimension (§5.2.1) that defines the architecture, the relationships and their semantics between the autonomous component. The organizational dimension (§5.2.2) allows us to define the roles of an autonomous component. The communication dimension (§5.2.3) specifies the nature of the relationships that exist between the autonomous components. The functional dimension (§5.2.4) defines the functions performed by autonomous component. The informational dimension (§5.2.5) describes the necessary information, especially the QoS information, for autonomic management.

5.2.1. The architectural dimension: the <Node, Link, Network> structure

The *NLN model* comes from the association of the meta-model with the concept of management levels. It provides a hierarchically structured object-oriented description of the real-world elements. In this representation, each Autonomous Component (AC) of "*network*" aggregates the ACs "*nodes*" and the ACs "*links*" that belong to its visibility level (horizontal integration), while it incorporates the behavior of ACs "*networks*" of lower visibility levels (vertical integration). Thanks to the integration of our QoS model (Perdigues, 1995), it becomes possible to automate QoS contracts. The SLA objective can indeed be derived across the different visibility levels, avoiding any useless redundancy of roles or responsibilities. A new paradigm, the *architecture engineering* (Nadour *et al.*, 2004), has been developed to carry out this vertical integration aspect of the architectural automation.

5.2.2. The organizational dimension: the policy-driven distribution model (PO)

In order to enable the network management to be flexible, we advocate organizing the managed system into several management-autonomous domains. A distributed Domain Management Decision Point (DMDP), by component of visibility level, has a great amount of autonomy to supervise and control the QoS. For example, on the network level, the Management Enforcement Point (MEP) monitors, controls and compares the QoS between its domain ingress points and egress points and performs backward actions. This network domain is defined by the "component" flow.

The network self-management requires local capacities of problem resolution for fast reactions. Managed objects, through their management interface, can be allotted, precisely, a role of autonomous agent delegated with enough intelligence and knowledge to make the decisions that have to be made. These decisions can be of a tactical nature, entrusted to DMDPs or MEPs, and derived from network-wide management policies held by the Management Decision Point (MDP), which reflects the management strategy of the network operator in order to let the MDP focus on strategic decisions. Following this description of the management role of networking component, we propose the Organizational Pilot (PO) (Nadour2 *et al.*, 2004) to enforce this organization. In each component of end-to-end service, there will be a PO acting as MDP in the user level. The same distribution of POs in the level of service and the only difference is the covered area is each service domain (DMDP). The PO which occupies a component acts as a MEP. Therefore, throughout the visibility level we designed, the border of AS domain should be taken into account and a PO acting as DMDP should be added in order to supervise and control the QoS as well as the MEPs in its domain. The dialogues between PO user level and PO service Level are managed by request and response in the client/server mode.

What's more, each PO should be able to handle the entity's QoS not only in every dimension (Availability, Reliability, Delay and Capacity), but also the different runtime, application related values as QoS conception, QoS threshold, and QoS current value. So, we can find in the figure 1, the class of Management is in fact the main functional part of PO which can realize all the management of the aspect of QoS. In the QoS conception information, there is the information derived from usage profile which describes constraints/possibilities allowing us to find the intersections to response a given context.

To help to implement this automation-friendly organization, we introduce in the next section a relational model to handle communication aspects. Note that the proposed model does not go as far as specifying the dialogues.

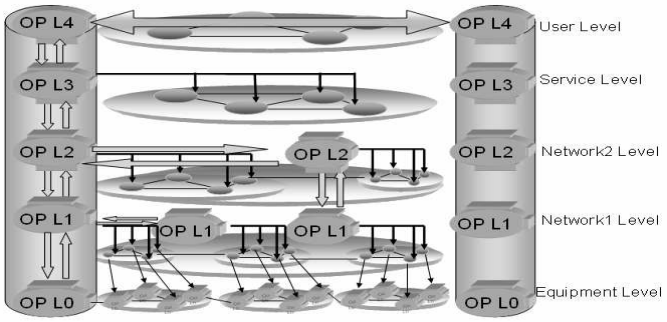


Figure 2: *A distributive management organization.*

5.2.3. The communication dimension: the relational model

According to their level of responsibility and in order to mutually perform the service, the distributed components interact and cooperate with each others through management relationships that bind them together. Besides the classical manager-to-agent {MDP -> MEP} subordination relationship, the relational model identifies the *peer-to-peer* cooperation relationship between autonomous agents (PO -> PO). A peer-to-peer relationship can be either a manager-to-manager relationship {DMDP/MEP - DMDP/MEP}, which reflects the cooperation between different domains, or a relationship between two agents (PO -> PO) to represent the cooperation between autonomous agents. The latter relation is more suitable for self-management as decisions are taken the closest to the managed components. This is the case for a QoS self-controlling node which would indicate to its neighbor node whether it is in or out of the contract. If not, the neighbor will be able to try to compensate.

5.2.4. The functional dimension

In the functional dimension, we distinguish a reaction aspect (5.2.4.1), with different possible reaction modes for autonomous agents, and a coordination aspect (5.2.4.2) which starts by the MDP, derived through the management levels, to be fully distributed and held by proactive and interactive autonomous agents (MDP/MDP).

5.2.4.1. Autonomic component reaction model

While interacting with other components to perform management tasks, an object can implement various types of behaviors. The reaction model identifies four possible reaction behaviors: passive, active, interactive and proactive.

When first introduced, the OP might have to dispatch by delegation the contracts of the QoS not only from the network which it is in charge but also from all other commands given by a management entity. When exploited, the PO acts both as “metrologist ” and controller. If the function is activated, it would then notify the failures to the entity that has the right. (*i*) In a *passive* role, the PO can only react autonomously to its behavior problems. (*ii*) In the *active* reaction mode, it can notify the other entities when QoS problems are detected. (*iii*) The *interactive* role is based on the active reaction mode but has the capability to interfere, which means in this context that it stands on an equal relationship with its counterparts. It acts as mediator in an inter-connexion in sub-networks (horizontal) or between two levels (vertical). (*iv*) As for the *proactive* role, which is also based on the mode of Interactive, possesses the knowledge and rules allowing it to make tactical decisions in order to target a problem in his own domains.

5.2.4.2. The automatic coordination model

The distributed components performing the considered service can be disseminated across several domains. Guaranteeing the end-to-end QoS requires consequently a closer harmonization between these distributed components. Precisely, the coordination model conducts the cooperative management process among the different managed domains. Since management decisions can be operational, tactical or strategic, the general organization of the coordination model may choose to delegate the operational decisions to LMDPs and the tactical ones to DMDPs in order to relieve the MDP which handles the strategic decisions. By laying down the management policies, a managed object will have the behavior adapted to each situation. These various behaviors correspond to the object status on its management interface.

5.2.5. The informational dimension: the information model

The information model represents the real-world entities into managed *Node* ACs, *Link ACs* and *Network* ACs of various levels. Thus, four types of MIBs (Daho *et al.,* 2004) are defined: (*i*) The Equipment-MIB (E-MIB) gives the representation of the network equipment. (*ii*) The Network-MIB (N-MIB) represents the image of the network level entities (flow, cross connections, ect.). (*iii*) The Service-MIB (S-MIB) represents the service components and its features. (*iv*) The Business-MIB (B-MIB) is associated to a specific provider's business objects. In order to support the information coherence, we propose to associate to each representative element, within each visibility level, a Finite State Machine (FSM). The associated FSM notifies the change of configurations, usage and behaviors (QoS) to all the entities concerned. An example of such FSM was given in (Daho *et al.,* 2003).

6. Application example : the Video on Demand service

In this section, we present an application example of our proposed models. This work was fulfilled in the VTC2e project "Virtual Telecommunication Community: every service for everyone" funded by our partner SFR. The example concerns the Video on Demand (VoD) service. It allows the customer to change the video language while he is watching his chosen video. To that end, we suppose that there is two video servers at least and can contain the same video with two different languages. The considered VoD servers and VoD client is given by the VL technology (http://www.videolan.org/) that proposes the Video LAN Client (VLC) and the Video LAN Server (VLS). We analyze the whole application according to four management levels. Within the service level, we find the VoD application constituted of service components (the VoD servers, the VoD Synchronizer and the VoD Client) bounded by virtual links. The leveled structure is obtained thanks to the application of our meta-model.

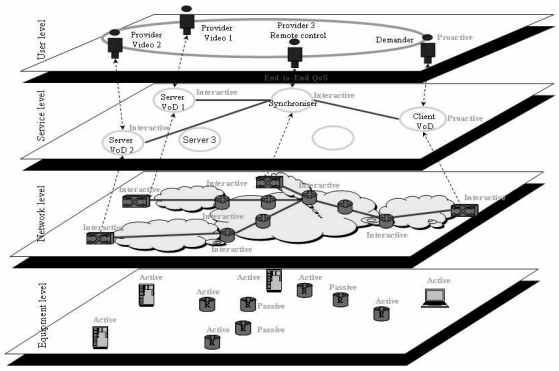


Figure 3: *The context of the VoD*

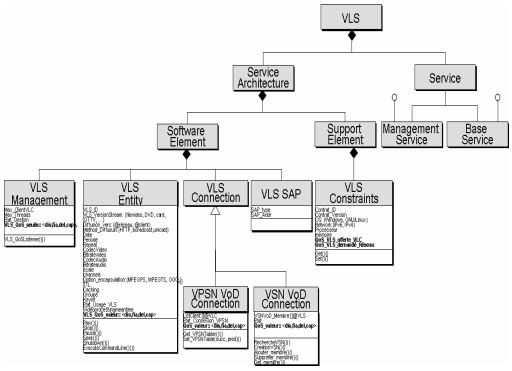


Figure 4: *The information model of the VoD Server*

6.1. The information model: the case of the VoD Server (VLS)

All service components of the VoD application have the same structure given by the <Node, Link, Network> information model. Figure 4 shows the structure of the VLS object that represents the VoD Server within the service MIB. In the VLS Entity class, QoS information is related to the current behavior of the VoD Server. The VLS Management class contains the necessary information to manage the VoD Server as well as the information related to the thresholds that limit its normal behavior. The VLS connection class contains QoS information related to the link between the VoD Server and the synchronizer. The VLS constraints class contains the local VoD server contract (local SLA) and the QoS that is requested to the network. Finally, the VLS SAP class represents the VoD Server access point.

6.2. The functional model: the case of the VoD Server

Within the service level, the behavior of the VoD servers is *interactive*. This means that the VoD servers are active entities that have the ability to interact with the other service components for exchanging the information before making a decision for example to switch the video flow to change the video language. This is translated within the information model by new class “interactive” in the service interface. The latter allows the VoD server to accept get and set requests to obtain for example the current sever capacity or to insert a new entry within the connection table of the VoD server. The VoD server can also send notifications to the synchronizer when current QoS values exceed the thresholds. These notifications allow changing automatically the video server to maintain the video QoS.

6.3. The organizational model

In our scenario, we distribute all the roles (Passive, Active, Interactive, and Proactive) to each PO who may participate in the simulation. The VoD demander situated in the user level, which can receive the requirement and collect the necessary information, which enables itself to make a decision and spread it to the inner level's PO, is given the role of Proactive. Similarly, the client VoD also has the role of Proactive. The three VoD servers have the roles of Interactive, because they are three equivalent peers considering the VoD service. Among the Interactive POs, they can exchange Peer-to-Peer information to cooperate with each other. So when we continue decline to the level of network we will also find the equivalent network routes have the roles of Interactive. The role of Active is that a PO can detect its own state and if there is anything going wrong, it can notify the manager.

6.4. The communication model: service requests of the VoD server

Within the service level, service components interact according to the VoD service logic to fulfill and to maintain the video service. Two kinds of interactions exist: Client/Server and Peer-to-Peer. (*i*) Client/Server interactions occur between the current VoD server in use, the synchronizer and de VoD client. (*ii*) Peer-to-Peer interactions occur between the service components that are functionally equivalent. In our example, the VoD Servers are functionally equivalent. Peer-to-peer relations reflect the cooperation between the VoD servers to maintain the video service QoS. For example, if a VoD server fails to respect its local SLA, it sends a notification to its neighbor that will be in charge of broadcasting the video stream.

6.5. The architectural model

This hierarchically structured description of the VoD application starts from the user level. This level is structured as a network of business entities that constitute the Virtual Private User Network (VPUN) (Rostambeik *et al.*, 2004). This network bonds the actors that participate to the fulfillment of the VoD service. The VPUN is based on the structure of the service level. This level is structured as a network of service components that constitute the Virtual Private Service Network (VPSN) (Daho *et al.*, 2006). This network bonds service components that constitute the VoD service. The VPSN is based on an IP VPN network. The latter network is based on equipment network constituted by equipment nodes bounded by equipment links. This architecture is obtained by using recursively the same <node, link, network> structure within each management level.

7. Conclusion and future work

With respect to the current and forthcoming challenges, autonomic networking is an attractive solution for operators. To achieve our objectives, we have first presented an overview of the self-managing proposals. The analysis of the current proposals shows that there is a lack in global framework to fully cover all the autonomic networking needs. We have presented the necessary models to support network and service autonomic management. We have shown the importance of the meta-model to address the entire problem. Our future orientations fall within methodological aspects to take into account all the presented concepts to effectively build integrated autonomic solution for next generation networks and services.

8. References

Boutabaa R., Xiao J., "Network Management: state of the art", Proceedings of IFIP World Computer Congress, 2002

Daho Z.B, Simoni N., "Towards Dynamic Virtual Private Service Networks: design and self-management", IEEE/IFIP NOMS'06, Vancouver, Canada 2006

Daho Z.B, Simoni N., "An information model for service and network management integration: from needs towards solutions", IEEE/IFIP NOMS'04, Korea, 2004

Daho Z.B, Simoni N., "Provisioning process: from customer needs to resource allocation", IEEE/IFIP LANOMS'03, Brazil, 2003

Kephart J.O., Chess D.M., "the vision of autonomic computing", Computer Society, 2003.

Martin-Flatin J.P., Znaty S., "A Survey of Distributed Enterprise Network and Systems Management Paradigms, JNSM, Vol. 7, No. 1, pp. 9-26, 1999.

Nadour T., Simoni N., Boutignon A., "Architecture Engineering: Mastering Architecture Evolution and Traffic Engineering Rules", ICC, Paris, Juin 2004

Nadour T., Simoni N., Chéné G., "Towards Dynamic Vertical Self-Organization", ICON, Singapore, November 2004.

Perdigues N., "Une approche pour définir des composants de service administrables : la tarification des nouveaux services'', Thesis dissertation, ENST-Paris, 1995.

Rostambeik S., Simoni N., "Userware: Virtual Service Access towards Usage Integration", NGNM, Networking 2004, Athens, Greece.

Sterritt R., "Autonomic Computing – a means of achieving dependability?" , IEEE Workshop on Engineering of Computer-Based Systems, 2003.

White S.R., Hanson J.E., "An Architectural Approach to Autonomic Computing", IEEE International Conference on Autonomic Computing (ICAC), 2004.

Building virtual organizations compliant with the ISO/IEC 17799 directives

Michel Kamel

Université Paul Sabatier
118 Route de Narbonne
F31062 Toulouse Cedex4 France
mkamel@irit.fr

Abdelmalek Benzekri, François Barrère, Bassem Nasser

Université Paul Sabatier
118 Rte de Narbonne
F31062 Toulouse Cedex04 France
{ benzekri,barrere,nasser}@irit.fr

ABSTRACT. *Virtual Organization (VO) has emerged as a type of collaboration between SMEs/SMIs to offer them new business opportunities by sharing resources and competencies. VOs, to function normally, necessitate the deployment of a secure shared IT infrastructure. ICT technologies are essential for VOs' foundation and improvement; they should be managed effectively to ensure the VO's success. An ISMS should be implemented to secure the Virtual Organization's information system. ISO/IEC 17799 and ISO/IEC 27001 are known as security standards and propose a way to model, implement and certify an ISMS. In this paper, we propose to apply ISO/IEC 17799 controls to Virtual Organizations in order to settle a secure infrastructure supporting this type of enterprise networks; we identify the ISO/IEC 17799 chapters we will use to achieve our objective.*

RÉSUMÉ. *L'Organisation Virtuelle (OV) est un type de collaboration entre PMEs/PMIs leur offrant de nouvelles opportunités métiers en partageant des ressources. Pour bien fonctionner, une OV nécessite le déploiement d'une infrastructure TI sécurisée. Les TICs sont essentielles pour le fondement et l'amélioration des OVs. Un SGSI doit être implémenté pour sécuriser le système d'information de l'OV. ISO/IEC 17799 et ISO/IEC 27001 sont reconnus comme des standards de sécurité ; ils permettent de modéliser, implémenter et certifier un SGSI. Dans ce papier, nous proposons d'appliquer ISO/IEC 17799 aux OVs pour mettre en place une infrastructure sécurisée supportant ce type de réseaux d'entreprises ; nous identifions les chapitres de l'ISO/IEC 17799 à utiliser pour réaliser notre objectif.*

KEYWORDS: *IT infrastructure management, information security, virtual organization, ISO/IEC 17799*

MOTS-CLÉS: *Gestion de l'infrastructure TI, sécurité de l'information, organisation virtuelle, ISO/IEC 17799*

1. Introduction

Internet and Information and Communication Technologies (ICTs) make possible the collaboration between enterprises and the sharing of resources to achieve common business goals; managing these technologies effectively is the only way to ensure the business continuity of these enterprises.

Today, many collaboration networks exist to let SMEs share resources. In this paper, we are interested in the Virtual Organization (or Virtual Enterprise) type of collaboration; a Virtual Organization (VO) is defined as a temporary network of independent enterprises with the purpose to fulfil a specific market requirement which can not be done when each enterprise works alone due to financial, technological or competencies lacks (Benzekri *et al*., 2005) (Thoben, 2004). A shared secure IT infrastructure should be deployed to support this type of collaboration networks. Managing the IT infrastructure is very critical to enterprises and a predominant need. For security management, international standards exist. The ISO/IEC 17799 standard (iso, 2005) allows the definition of the Information Security Management System (ISMS) but does not say how to implement it; it is ISO/IEC 27001 (iso, 2005) that does it and allows its certification. These standards are used actually to model and deploy ISMS for single enterprises and approach the third party access concept.

Dependence on information systems and services means organizations are more vulnerable to security threats. Many scenarios could be utilized to highlight the relation between users and enterprises and how these enterprises could afford access to these users to their information systems without risking their business ruin. The interconnection of public and private networks (project zones or sites) and sharing of information resources increases the difficulty of achieving access control. The trend to distributed computing has weakened the effectiveness of central, specialised control. In this paper, we are interested in how to build a VO based on a secure IT infrastructure through the definition of security controls to implement accordingly to the ISO/IEC 17799 standard.

This paper starts by introducing the Information Security concept and its benefits for the enterprises in general. In section 3 we introduce the ISO/IEC 17799 standard; its contents as directives and controls. In section 4, two scenarios of interaction between users and enterprises are given to introduce the concept of access management within SMEs; these two scenarios will let us think on a more complex scenario which is the management of access between SMEs developed in section 5. Section 6 enumerates some related works. In section 7 we highlight on the ISO/IEC 17799 chapters that we will use to implement the security controls necessary for a VO foundation. Finally we introduce the security practices' maturity concept which evaluates the compliance of any SME with the ISO/IEC 17799 standard and conclude with future works.

2. Information security: definition and benefits

Information is an asset that has value to an organization and consequently needs to be suitably protected. Whatever form (paper, electronic, etc.) the information takes, or means by which it is shared or stored, it should always be appropriately protected. Information security protects information from a wide range of threats in order to ensure business continuity, minimize business damage and maximize return on investments and business opportunities. Information security is achieved by implementing a suitable set of controls, which could be policies, practices, procedures, organizational structures and software functions. These controls need to be established to ensure that the specific security objectives of the organization are met. Information security is characterized as the preservation of:
- Confidentiality: protecting sensitive information from unauthorized disclosure. Basically it is the assurance that information is not disclosed to inappropriate entities or processes;
- Integrity: detecting whether there has been unauthorized modification of data. It ensures that data is the same as that in the source documents and has not been exposed to accidental or malicious alteration or destruction;
- Availability: ensuring that authorized users have access to information and associated assets when required.

Information security management needs, as a minimum, participation by all employees in the organization. It may also require participation from suppliers, customers or shareholders. In the next paragraphs, we will be interested in ISO/IEC 17799 as international standard for information security management.

3. ISO/IEC 17799

ISO/IEC 17799 is the code of practice for information security management. ISO/IEC 17799 lays out a well structured set of controls to address information security risks, covering confidentiality, integrity and availability aspects. It was first published in December 2000. An enhanced version of ISO/IEC 17799 appears in late 2005; it contains eleven main sections specifying 39 control objectives and 133 security controls. A *control objective* can be defined as a statement of the desired result or purpose to be achieved by implementing control procedures within a particular IT activity. A *security control* can be defined as the policies, procedures, practices and organisational structures designed to provide reasonable assurance that business objectives will be achieved and that undesired events will be prevented or detected and corrected. The ISO/IEC 17799:2005 eleven core chapters are:

- Security policy

Provides guidelines and management advice for improving information security.

- Organizing information security

Facilitates information security management within the organization.

- Asset management

Carries out an inventory of assets and protect these assets effectively.

- Human resources security

Minimizes the risks of human error, theft, fraud or the abusive use of equipment.

- Physical and environmental security

Prevents the violation, deterioration or disruption of industrial facilities and data.

- Communications and operations management

Ensures the adequate and reliable operation of information processing devices.

- Access control

Controls access to information.

- Information systems acquisition, development and maintenance

Ensures that security is incorporated into information systems.

- Information security incident management

Defines a plan to act when incidents occur and disturb the right operation of the security system.

- Business continuity management

Minimizes the impact of business interruptions and protect the company's essential processes from failure and major disasters.

- Compliance

Avoids any breach of criminal or civil law, of statutory or contractual requirements, and of security requirements.

The ISO/IEC 17799 standard covers technical, administrative and legal aspects; it gives recommendations and directives on how to secure an enterprise information system by modelling the information security management system needed. The controls to deploy within any SME are chosen depending on the definition of security requirements through risk assessment and risk analysis methods. To implement and certify the ISMS needed, ISO/IEC 27001 is used. The ISO/IEC 27001 management standard, entitled "Information Security Management - Specification with Guidance for Use", instructs the enterprise's security responsible how to apply ISO/IEC 17799 and how to build, operate, maintain and improve an ISMS using a continual improvement approach known as the Plan-Do-Check-Act model.

4. Scenarios of access management within SMEs

To protect its information assets, an enterprise could decide to deny any access to them. Although this solution is a valid one, it is not always applicable. It is clear that all enterprises, in some circumstances, are obliged to manage users' accesses from inside, or outside, their boundaries. Two scenarios are adopted to represent the relations between users and enterprises.

4.1. *Visitor's access*

The first scenario is about a user, having the visitor status, wanting to access the internet from the enterprise's site. The IT administrators must consider, within the enterprise global security policy, to authorize users to access the web using their mobile stations (laptop, PDA, etc.) through the enterprise's infrastructure. To allow its visitors access the internet, an enterprise could afford them a LAN access or a WIFI access. In the two cases, the enterprise's policy must take into consideration one primary security service which is "availability"; in other words, the user must be authorized to access the web anytime he wants but he will not be able to do anything else. Whether it is a LAN access or a WIFI access, the user must be authenticated by the enterprise authentication server before it is authorized to access the web. In the two cases and for a robust authentication, the EAP (extensible authentication protocol) can be used: EAPoL" (EAP over LAN) for LAN accesses and EAPoW (EAP over Wireless) for wireless accesses (Latzer, 2002).

A management process is defined for users' accesses; it depends on the enterprise policy. When a user (enterprise visitor) asks for an internet access, he will have an account by simply providing the IT administrator a proof of identity (passport, credit card, etc.). The IT administrator creates an account for him and defines the credentials he has (access the web) within the enterprise directory server; the user receives a token, a connection ID and a password, or a digital certificate, etc. The user must accept the enterprise's charter. When the user attempts to access the internet, by connecting its workstation to a local port or through a wifi access, the authenticator (switch or access point) asks the supplicant for an identity. The user enters its connection ID and password, or uses its token or sends its digital certificate; it depends on the authentication method used over EAP. The authenticator relays the authentication elements to the authentication server (ex: RADIUS) which has to validate the identity of the user by requesting the directory server (ex: LDAP) for the user credentials. The IP address for the connection may be received from the DHCP server dynamically or reserved by the IT administrator.

Once the user is authenticated, he is allowed to access the internet and this is defined through the access control policies implemented within the enterprise's IT infrastructure (within the directory). The IT enterprise administrator must have implemented audit and log tools for traceability and filing; a policy for log and audit should be deployed within the global security policy. Examples of security policies to implement:

- To manage user access, a policy is implemented:

Access to the resources on the network must be strictly controlled to prevent unauthorized access.

- To ensure the availability security service, these policies are implemented:

All equipments owned by the organization must be supported by appropriate maintenance facilities from qualified engineers.
All equipments should be protected from power failures.

- To ensure compliance with the enterprise security policy, this policy is implemented:

All employees are required to fully comply with the organization's information security policies. The monitoring of such compliance is the responsibility of management.

4.2. *Nomadic user's access*

The second scenario is about nomadic users. Enterprises, in many circumstances, are obliged to allow external users (clients or employees) to access their IT infrastructures in order to share or extract data, although that could cause harmful attacks. External accesses must be managed according to the enterprise's global security. Many solutions could be used to allow users to access the enterprise's IT infrastructure securely: an L2TP over IPsec VPN solution or an SSL-based solution. The first solution will be detailed in the next paragraphs. An L2TP over IPsec VPN solution (cf. figure 1) consists in configuring an L2TP connection for remote access client over an IPsec connection; L2TP is used to provide user authentication and IPsec provides computer authentication and L2TP tunnel encryption.

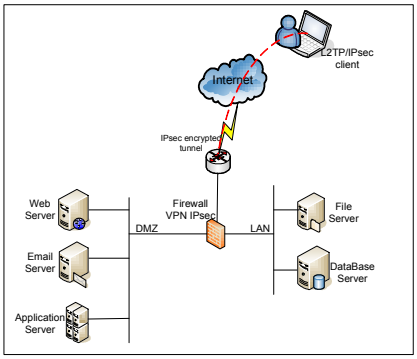


Figure 1. *A VPN L2TP/IPsec solution for external accesses*

Confidentiality and integrity security services must be ensured to allow users to access the enterprise's infrastructure securely; asymmetric cryptographic systems based on "private" and "public" keys should be used. Public keys must be linked to physical identities through digital certificates which allow ensuring that an entity is really the one it claims to be. A *digital certificate* is an electronic "credit card" that establishes user credentials when doing business or other transactions on the Web. It contains the user's name, a serial number, a copy of the certificate holder's public key (used for encrypting messages and digital signatures), and the digital signature of the certificate-issuing authority so that a recipient can verify that the certificate is real. A digital certificate is issued by a Certification Authority (CA). A Certification

Authority is an element of a total *Public Key Infrastructure* (PKI) (ietf, 2005) implemented to ensure the management of digital certificates; we mean by digital certificate's management the creation, the signing and the validation of the digital certificates exchanged between parties to prove their identities.

A management process should be defined in order to let only the authorized users access the enterprise information system after being authenticated. Each user wanting to access the enterprise resources from everywhere through the web must have an L2TP and an IPsec clients on its workstation. The IT administrator must implement a VPN server on the enterprise site in order to receive the requests from the VPN clients. The IT administrator sends a couple (a username and a password) to the user and configures the L2TP server to accept the connection of the user providing this couple. Then he must configure IPsec on the server to authenticate IPsec clients through digital certificates or a pre-shared key. If a solution based on pre-shared key is used, the key must be known by the two parties. If a solution based on digital certificates is used, the user must have a digital certificate signed by an authority which is the one that signed the server certificate or has confidence in it. The user configures an L2TP/IPsec VPN connection; when the VPN client attempts to connect to the VPN server, he sends its digital certificates (if it is a solution based on certificates), then enters its username and password. The VPN server authenticates the user and an IPsec encrypted tunnel is built between the two parties. The data exchanged between them over the IPsec tunnel will be encrypted to ensure data confidentiality and integrity.

Deploying an L2TP/IPsec VPN solution necessitates taking into consideration legal aspects related to the use of encryption tools: using these tools to encrypt the exchanged data necessitates authorization from legal communities. Enterprises should refer to the encryption technology laws specific to their countries. When implementing its cryptographic policy, an enterprise must consider the regulations and national restrictions that might apply to the use of cryptographic techniques in different parts of the world and to the issues of trans-border flow of encrypted information.

The user can only do what he is authorized to do; this is defined within the security policies on the VPN server side. Access control lists should be defined and implemented on the VPN server to control the access to the enterprise local network. Audit and log tools should be installed for traceability; intrusion detection systems should also be installed to detect any unauthorized access. In the global security policy of the enterprise, policies for access control and user's responsibilities must be explicitly defined. Examples of security policies to implement:

- To manage user access, a policy is implemented:

Access to all systems must be authorized by the owner of the system and such access, including the appropriate access rights must be recorded in an access control list.

- And for a robust security, this policy also is implemented:

Access controls are to be set at an appropriate level which minimises information security risks yet also allows the organization's business activities to be carried without undue hindrance.

These two scenarios let us think on a more complex scenario as site-to-site or project zone-to-project zone connections in order to share resources and collaborate on business projects; this concept is used to build what is called Virtual Enterprises (VEs) (Thoben, 2004).

5. The Virtual Enterprise concept

We will detail the concept of project zone-to-project zone interconnection as it allows to collaborate on specific projects and to place, at the disposal of our partners, specific resources and not all the enterprise infrastructure's resources.

Here, we introduce the concept of Project Zones (cf. figure2). A Project Zone is a zone within an enterprise having its own project specific security measures. The Project Zone is only accessible for project trusted Entities (Benzekri *et al*., 2003).

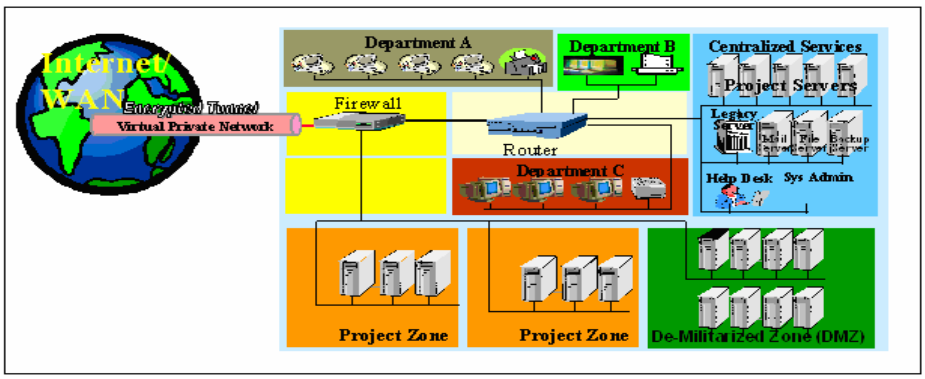


Figure 2. *The Project Zone concept*

Resources within the project zone will be separated from the other ones as if they were on another place. To access these resources, solution like L2TP/IPsec VPN could be used to provide the confidentiality and the integrity of the information exchanged between the two sites. To implement such solution, we must take into consideration these principles:

- An address plan should be adopted

Concerning the address plan, we can consider a solution which is to specify an address space for the Virtual Enterprise (connection of the two project zones) and burst it into two spaces and affect a space to each one of the project zones.

- A routing plan should also be adopted

Concerning the routing plan, the routing tables should be changed (add new entries) to take into consideration the new routes that should be borrowed to reach destination (on the other site); this allows the VPN server on each site to consider the new address plan (specific to the project zone).

For domain name resolution, as we want to have enclosed project zones in order to secure the resources deployed within them, a DNS server should be installed in each project zone. The L2TP/IPsec VPN solution necessitates the utilization of digital certificates to ensure a more secure share of data as they are the most valuable proof of identity especially when they are exchanged within a public key infrastructure (Hastings *et al*., 2001). We can consider two ways to implement a PKI solution for digital certificates distribution:

- A Public Key Infrastructure per site

On each site, a public key infrastructure should be implemented to create, distribute and validate digital certificates given to end points on the site. A cross-certification between the two CAs is necessary. These two CAs should be reachable over the web in order to validate the certificates and request the certificates revocation lists; they must be located within the DMZ zones of each site. Such a solution is most suitable if the two PKI domains belong to two work contexts that share a close working relationship with each other. Its disadvantages are complexity and poor scalability.

- A PKI accessible through the web

In this case, a unique PKI, accessible from everywhere through the internet, is implemented (cf. figure 3). The CA within this PKI must create digital certificates to associate public keys to identities for each end point on the two sites where the project zones are defined. Advantages of such a solution are: reduction of the administrators' management responsibilities, high scalability, and simplicity of the certification process.

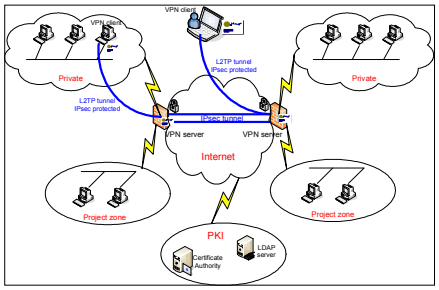


Figure 3. *The Virtual Enterprise project zones*

The many solutions evoked for PKI, DNS, routing rules, VPN should be implemented within the enterprise's infrastructure as a unit and as part of the global enterprise's security policy which should be improved continually each time risks

and threats vary on the enterprise's information assets. Information and competencies sharing is beneficent to SMEs and to their business improvements, but it may cause the loose of confidential and vital information when the enterprise IT infrastructure is not managed effectively. The security issues that must be considered when connecting project zones are (Jaiswal, 2004):

- Authentication & Access Control

Authentication is being able to prove that a user or application is genuine. To enforce authentication, a Single Sign-on (SSO) solution could be used.
Access control is the process of protecting critical resources in a system.

- Intrusion Detection/Prevention

Intrusion Detection/Prevention is the act of detecting and preventing from unauthorized accesses.

- Security auditing and management

Security Auditing is an independent review and examination of data processing system records and activities to test for adequacy of system controls, to ensure compliance with the established security policy and operational procedures, to detect breaches in security, and to recommend any indicated changes in control, security policy, and procedures.
Security Management is the management of an organization's security policy by monitoring and controlling security services and mechanisms, distributing security information, and reporting security events.

- Non-repudiation

The *Non-repudiation* concept involves a mechanism that is used to prove the identity of involved in a communication or having participated in all or part of the communication. It is an extension of authentication, but is usually used for legal purposes.

6. Related works

Nowadays, SMEs wishing to secure their information systems whenever they authorize external users to access their resources, may adopt audit methods such as EBIOS (Ebios, 2005) and MEHARI (Mehari, 2005). EBIOS focuses on defining security needs through risk assessment but does not recommend security practices to implement on the enterprise level. Using such a solution is beneficent but not sufficient. The ISO/IEC 17799 standard offers the SMEs administrators the possibility of evaluating the implemented security practices by referring to the standard's recommendations which gives the evaluation a sense on an international level and leads to an ISO/IEC 27001 certification. Nowadays, SMEs wishing to found a VE, refer to solutions or technologies in order to ensure the security of the resources exchanged; these approaches could be based on trust relationships or contracts, business reputation, business objectives, etc. The mission of the TrustCom project (Tuptuk et al., 2004) is to provide a trust and contract management framework enabling the definition and secure enactment of collaborative business

processes within VEs. Until now, the approaches adopted don't take into consideration the foundation of a VE compliant with a standard's directives in order to model and implement a management system for the distributed information system. This is why we have dealt with it in this paper

7. Applying ISO/IEC 17799 standard to Virtual Enterprise

After defining the security requirements for the project zones to form what is called Virtual Enterprise, we identify the security controls to be implemented. The IT administrators must deploy these controls, accordingly, and as a unit, to reduce security breaches that could be used by hackers to access the VE's information system. In the next paragraphs, we detail the necessary chapters for our approach and the control objectives to realize in order to build a secure VE.

- Security policy chapter
- *Information security policy* objective

The Virtual Enterprise must define a comprehensive information security policy, consistent with its business objectives, meeting its business requirements and complying with laws and regulations. The VE information security policy is an extension of the information security policy of each of the SMEs. It is not intended to modify or render invalid what has already been established. A VE information security policy must reflect its business needs and be accepted by the group of partners.

- Organizing information security chapter
- *Internal Organization* objective

The implementation of the enterprise's global information security policy should be controlled through the deployment of a management framework within the IT infrastructure. Security roles (project manager, referents and administrators) are defined and affected to persons within the enterprise network to form the responsibility chain.

- *External Parties* objective

The *external parties* control objective to be achieved, external accesses should only be authorized if they do not expose the VE's resources to harmful attacks. The VE IT managers must maintain the security of their enterprise whenever there is access from external parties or when dealing with customers. In VE, management processes should be applied dynamically to face the fast changes of the enterprise security policy (users leave, users arrive, security requirements vary, etc.).

- Communications and operations management chapter
- *Network security management* objective

Special policies should be defined to protect the information on the VE's network, the infrastructure that supports the network itself and sensitive data that should pass over public networks. Routers, firewalls, proxies, NIDS must be deployed and

configured to protect the VE's network; VPN solution is adopted to ensure the protection of sensitive information exchanged through the web.

- *Exchange of information* objective

Exchange of information and software between partners within the VE or between the VE's sites and external parties must be protected and controlled through formal agreements.

- *Monitoring* objective

As information system problem could occur, the VE IT administrators must use operator and fault logs to detect these problems; monitoring information processing systems allows the detection of unauthorized activities. The system should also be monitored to check the effectiveness of controls applied, and to verify that information processing activities comply with the VE's access policy.

- Access control chapter
- *Business Requirements for Access Control* objective

As access control must be driven by business needs, the VE IT administrators must define an access control policy to control access to information and to business processes. Access control rules should comply with information authorization policies. Access control models exist; Or-BAC (Organization Based Access Control) (Abou El Kalam *et al*., 2003) (Nasser *et al*., 2005) can be used to express the access control rules. Or-BAC is adapted to Virtual Enterprises (Organizations), as it is proposed for modelling a security policy that is not restricted to static permissions.

- *Network access control* objective

Users must be provided with an "access only" right to the services that they are authorized to use. For remote users, the IT administrators must define the authentication mechanism (digital certificates, tokens, passwords, etc.) to control their accesses. Requirements for physical protection, access controls, cryptographic techniques and virus protection should be included in the VE policies. As segregation between networks is adopted; security mechanisms such as routers and firewalls are deployed. Filtering and routing rules must be defined and deployed on the edge routers to control access from, and towards the enterprise's information system.

- Compliance chapter
- *Compliance with legal requirements* objective

The VE managers must make sure that the VE's information system complies with statutory, regulatory and contractual security requirements. Depending on the country laws for securing information, cryptography techniques are chosen.

8. Security practices' maturity

When deciding to found a VE, SMEs wishing to collaborate through the shared infrastructure, have to bring, or use, resources and competencies to their partners. The totality, or a part, of their information system will be part of the VE distributed

information system. In this paper, we consider a solution based on the interconnection of project zones to form a VE; in this case, SMEs must deploy project zones within their infrastructures. Questions to be raised are: has this SME the competencies to deploy project zones securely? Are management roles (IS managers, IT security manager, etc.) defined within it to ensure the right implementation of security policies? Has this SME implemented practices that ensure the security of its information assets? ISO/IEC 17799 gives SMEs recommendations for information security management by enumerating the security practices allowing them achieving their control objectives. We have to develop a tool to measure the conformity of the security practices deployed within the SMEs with the ones enumerated by the ISO/IEC 17799 standard; in other terms, the tool will measure the maturity of the enterprise security practices. We hope to experiment it in practical business cases dealing with the VIVACE (Value Improvement through a Virtual Aeronautical Collaborative Enterprise) project. Depending on the level of maturity of these practices, the VE enterprise project manager will decide how to deploy the VE and what will be the role of each SME in order to avoid putting in danger the VE's resources. This subject is out of the scope of its article and it will not be treated here.

9. Conclusion and future works

Virtual Organizations collaborative networks allow SMEs sharing resources and competencies which lead to new business opportunities. Collaboration between enterprises through IT infrastructures is beneficent for them but it does also present risks. In this paper, we have presented scenarios for access management that SMEs can face when dealing with external and internal users; then we have widen the scope to take into consideration more complex scenarios as the interconnection of project zones to form the virtual organization network. In order to settle a secure infrastructure supporting a virtual organization, we propose to use the ISO/IEC 17799 security management standard that helps VE managers to deploy the necessary controls to ensure the security of the distributed information system. The next step is to realize the tool that evaluates the maturity level of the SMEs' security practices; we will also be interested in how to enable trust between SMEs within a VE. Within our next studies, we will be interested in the concepts that allow us to implement on-demand and self-management virtual organizations based on a secure IT infrastructure.

10. References

1. Abou El Kalam A., El Baida R., Balbiani P., Benferhat S., Cuppens F., Deswartes Y., Miege A., Saurel C., Trouessin G., "Organization Based Access Control", ENST, 2003

2. Benzekri A., Barrère F., “Recommendations for network reliability and security for collaborative working in aeronautical SMEs”, CASH project, Deliverable D3.2, D32V2.080403.IRIT.CASH, Toulouse, 2003

3. Benzekri A., Kamel M., Barrère F., “Virtual Organizations Infrastructure and Team”, VIVACE project, Deliverable D0.3.3_x, Issue n° 0.1, Toulouse, 2005

4. Ebios. http://www.ssi.gouv.fr/fr/confiance/methodes.html, 2005

5. Hastings N., Polk W., “Bridge Certification Authorities: Connecting B2B Public Key Infrastructures”, NIST, 2001

6. ISO/IEC information centre. http://www.standardsinfo.net/isoiec/index.html, 2005

7. Jaiswal R., “Security concerns in EAI”. Wipro Technologies, 2004

8. Latzer T., “Wireless LAN Security”, Cisco systems, 2002

9. Mehari. https://www.clusif.asso.fr/fr/production/mehari, 2005

10. Nasser B., Benzekri A., Laborde R., Barrère F., Kamel M., “Access Control Model for Inter-organizational Grid Virtual Organizations”, IRIT, 2005

11. Public-Key Infrastructure (X.509) (pkix). http://www.ietf.org, 2005

12. Thoben KD., “The Networked Supply-Chain. IST-2001-65001 IMS-NoE – A project of the FP5 IST Programme”, Deliverable NO. D17-18 Annex, 2004

13. Tuptuk N., Lupu E., “State of the art”, TrustCom project, Deliverable2, Issue 1, Imperial College London, 2004

SESSION 5
LA SECURITE

Apprentissage de nouvelles attaques avec un modèle de Case-Based Reasoning

Karima Boudaoud, Nicolas Nobelis

Laboratoire I3S-CNRS,
Université de Nice Sophia-Antipolis,
Bat. EPU, 930 route des Colles, BP 145
06903 Sophia Antipolis Cedex
{karima, nobelis}@essi.fr

RÉSUMÉ : *Un système de détection d'intrusion nommé AMASIR (A Multi Agent System for IntRusion detection), géré par des politiques de sécurité et fondé sur un système multi-agents, a été proposé pour détecter des attaques connues. Dans ce papier, nous proposons d'étendre le système AMASIR avec un model de Case-Based Reasoning (Raisonnement à base de cas) pour apprendre de nouvelles attaques.*

ABSTRACT : *An intrusion detection system named AMASIR (A Multi Agent System for IntRusion detection), driven by security policies and based on a multi-agents system has been proposed to detect known attacks. In this paper, we propose to extend the AMASIR' system with a Case-based Reasoning model to learn new attacks.*

MOTS-CLÉS : *détection d'intrusion, apprentissage, Case-Based Reasoning, politiques de sécurité*

KEYWORDS : *intrusion detection, learning process, Case-Based Reasoning, security policies*

1. Introduction

Chaque jour, les réseaux d'entreprises sont menacés par divers types d'attaques complexes. La sécurité de ces réseaux est gérée via deux approches principales : une approche préventive, qui consiste à protéger les données et les identités, et une approche de détection, qui consiste à surveiller en permanence les comportements et les états suspicieux. Par exemple, le cryptage des données, le déploiement de pare-feux ou l'authentification des utilisateurs par mot de passe appartient à la première famille de mécanismes de sécurité alors que l'utilisation d'un système de détection d'intrusion (IDS) appartient à la seconde. Les systèmes de détection d'intrusion ont prouvés être une manière efficace de détecter les attaques survenant dans un réseau. Cependant, tous les IDS font face au même défi : comment détecter de nouvelles attaques ?

Un système de détection d'intrusion nommé AMASIR (A Multi Agent System for IntRusion detection), géré par des politiques de sécurité et fondé sur un système multi-agents a été proposé [1], [2]. Dans ce système de détection d'intrusion, des agents intelligents coopèrent et communiquent en eux pour détecter efficacement des attaques en accord avec les schémas d'attaque enregistrés dans leur base de connaissance. Les caractéristiques clés des Agents Intelligents (AI) utilisés dans ce système sont la délégation, la coopération et la communication. Cependant, une propriété importante des AI, l'apprentissage, n'a pas été utilisée. Ainsi, cet IDS n'a pas la possibilité de détecter de nouvelles attaques. C'est pourquoi nous proposons d'étendre ce système de détection d'intrusion fondé sur des agents en ajoutant une fonctionnalité d'apprentissage de nouveaux schémas d'attaque. Cette fonctionnalité va permettre de mettre à jour la base de connaissance d'AMASIR quand un nouveau schéma d'attaque sera découvert.

En général, dans les IDSs existants, la notion d'apprentissage est utilisée pour apprendre les comportements normaux du système à sécuriser. Pour cela, dans une phase dédiée d'apprentissage, des profils normaux sont construits. Ces profils sont ensuite comparés avec l'activité en cours. Dans notre cas, nous pensons qu'il serait plus intéressant d'utiliser cette notion pour apprendre les comportements anormaux qui correspondent aux attaques. Pour apprendre une nouvelle attaque, nous devons d'abord la détecter et ensuite mettre à jour la base de schémas d'attaque. Pour la phase de détection, il est nécessaire de sélectionner une approche de détection adéquate. Le Data Mining (DM) semble à l'heure actuelle une des techniques les plus populaires parmi celles visant à découvrir des corrélations et des structures significatives et non intuitives dans un large jeu de données [3]. Cette technique est appliquée dans les deux grandes catégories de détection d'intrusion, l'approche comportementale et l'approche par scénarios. Dans la première, chaque élément du jeu de données est classifié et libellé comme une activité normale ou anormale, ce qui permet de générer des règles servant à trier facilement de nouvelles données [4], [5], [6]. Dans la seconde approche, un jeu de donné est enregistré durant une utilisation normale du système et est conservé comme cas de référence pour le

comparer à l'activité en cours et détecter des attaques potentielles [7], [8]. Dans notre cas, nous ne voulons pas construire un profil normal avant la phase de détection; c'est pourquoi le DM n'est pas approprié pour construire notre modèle d'apprentissage. Une autre technique nous semble plus appropriée pour répondre à nos besoins : Il s'agit du Case-Based Reasoning (CBR) [9], [10], [11]. Dans ce contexte, nous proposons d'utiliser cette technique pour permettre aux différents agents d'AMASIR de découvrir des similarités entre les attaques passées et les nouvelles attaques.

Notre papier est structuré de la manière suivante. Nous commencerons par donner une vue d'ensemble du système de détection d'intrusion AMASIR. Ensuite, nous présenterons notre approche d'apprentissage ainsi que des systèmes de Case-Based Reasoning existants. Puis nous décrirons notre modèle d'apprentissage et nous nous focaliserons sur l'algorithme de CBR, en particulier la fonction de similarité. Finalement, nous présenterons rapidement l'implémentation et nous conclurons avec quelques remarques qui préfigureront nos travaux futurs.

2. AMASIR: Un IDS multi-agents

AMASIR est constitué de différents types d'agents structurés hiérarchiquement. Une description détaillée est donnée dans [1], [2]:

- **L'administrateur** : il s'agit typiquement d'une personne qui sélectionne et fournit les politiques pour un réseau, ce qui inclus les politiques de détection générales, les politiques de réponse, et les nouveaux schémas devant être appliqués.
- **L'agent gestionnaire de politique de sécurité** : cet agent interagit avec l'administrateur pour recevoir des politiques, et traduit ces dernières en schémas d'attaque, qui sont ensuite transmis à *l'agent gestionnaire de réseau*.
- **L'agent gestionnaire extranet** (ou de réseau) : cet agent reçoit les schémas d'attaque à détecter sous forme de buts, qui sont passés sous forme de schémas d'attaque aux *agents gestionnaires intranet*.
- **L'agent gestionnaire intranet** : cet agent reçoit des schémas d'attaque, qui sont traduit en buts de détection pour agents individuels. Cet agent transmet également ses suspicions à son agent gestionnaire de réseau et à ses pairs, les autres *agents gestionnaires intranet*.
- **L'agent surveillant local** : cet agent surveille un système en accord avec les buts donnés par son agent gestionnaire. Il communique avec ses pairs (i.e. les autres agents surveillants locaux) pour confirmer ses suspicions, et, si ces dernières deviennent des croyances d'une attaque, elles sont reportés à son *agent gestionnaire*.

Dans le modèle informationnel d'AMASIR, plusieurs éléments ont été définis : *les messages, les événements, les filtres d'événement, les séquences d'événement* et *les schémas d'attaque* [1], [2].

Les messages représentent les messages réseaux (message ICMP, message TCP, etc...) et les messages systèmes qui sont collectés respectivement du trafic réseau et des journaux systèmes.

Un événement sécurité est défini par un critère de sélection qui concerne les attributs d'un message, qui peut être :

- simple ou mono-message pour un critère de sélection concernant les attributs d'un message (tel que source et/ou destination d'un paquet réseau)
- multiple ou multi-message pour un critère de sélection concernant plusieurs messages.

Un événement sécurité est caractérisé par son type, son point d'observation, un attribut temporel (représentant le moment auquel l'événement survient) , et un set d'attributs atemporels. Suivant le type de l'événement et son point d'observation, nous identifions une classe d'événement principale, *GenericSecurityEvent*, et huit sous-classes d'événement [1] (La fig.1 montre ces classes).

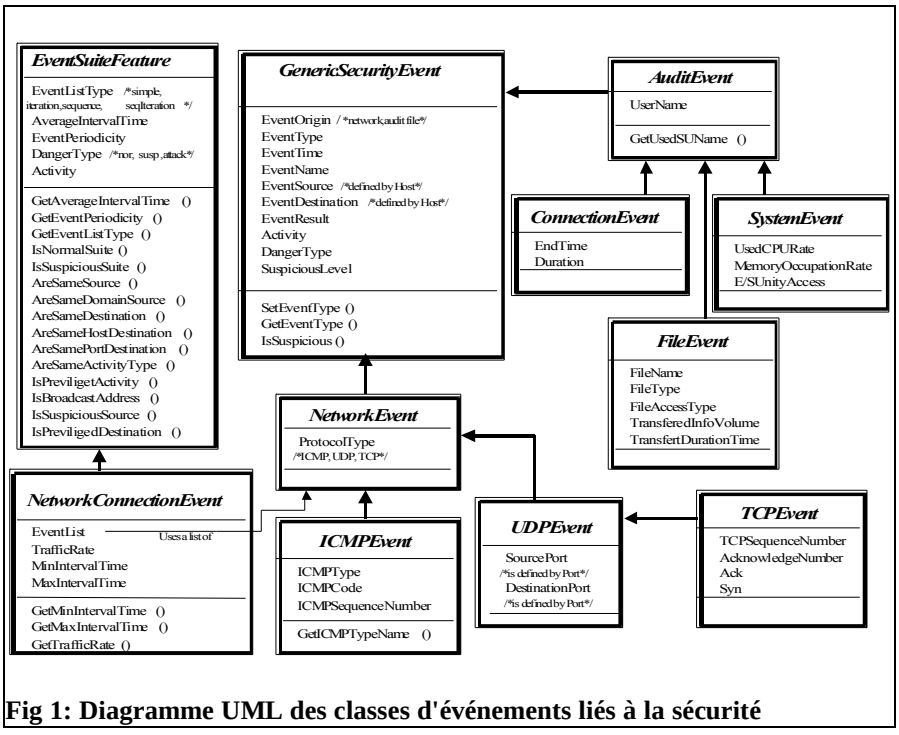


Fig 1: Diagramme UML des classes d'événements liés à la sécurité

Un *filtre d'événement* permet la sélection de messages pertinents pour obtenir les événements devant être analysés. Il spécifie le critère de sélection mono et multi-message. Il caractérise également les événements à filtrer en spécifiant :

- les valeurs des attributs concernant essentiellement le point d'observation, le type et d'autre attributs atemporels (comme la source et/ou la destination des événements),
- les opérateurs de comparaison sur ces valeurs d'attributs. Ces opérateurs spécifient le critère de sélection de ces événements. Pour décider si un événement doit être collecté ou non, une évaluation des différents opérateurs est effectuée.

Les *séquences d'événement* représentent un sous-ensemble des événements de sécurité extraits de l'ensemble des événements collectés dans une fenêtre d'observation temporelle. L'extraction de ce sous-ensemble est effectuée en utilisant un critère de sélection de séquences d'événement. Ce critère permet une agrégation des événements d'une même séquence. Cette agrégation concerne, par exemple le nom d'utilisateur, l'adresse source, l'adresse destination.,...

Les *schémas d'attaque* 1) définissent les filtres d'événement et 2) regroupent les événements collectés en séquences d'événement. Un schéma d'attaque est une classe qui caractérise une sorte d'attaque de sécurité. Une instance de schéma d'attaque est une description d'une attaque paramétrée. Afin de décrire un schéma d'attaque et de l'instantier, un ensemble d'opérateurs a été défini pour :

- fixer des conditions et donner des valeurs aux attributs des événements,
- fixer des contraintes temporelles et des critères d'agrégation d'événements,
- reconnaître des similarités entre une séquence d'événement et un schéma d'attaque.

Pour détecter efficacement des attaques, les agents d'AMASIR combinent des capacités cognitives (fondées sur la connaissance), permettant de raisonner sur des attaques complexes, avec des capacités réactives (stimulus-réponse), permettant de réagir rapidement quand des événements indiquent un état anormal du réseau (e.g. une congestion du réseau due à une attaque de type déni de service). Ainsi, un agent a trois fonctions : *le filtrage d'événement*, *l'interaction* et la délibération.

La *fonction de filtrage d'événement*, qui existe seulement dans les *agents surveillants locaux*, filtre les événements de sécurité pertinents s'étant produits dans l'environnement de l'agent, suivant les classes d'événements spécifiées à l'agent quand il reçoit un but de détection. En fait, quand l'administrateur spécifie les politiques de sécurité, un ensemble de buts de détection à atteindre est dérivé de ces politiques (par l'agent gestionnaire de politique de sécurité). Ces buts sont crées avec les instances de schémas d'attaque à détecter et sont envoyés aux agents appropriés : dans un premier temps à l'agent gestionnaire réseau, ensuite à l'agent

gestionnaire intranet puis enfin aux agents surveillants locaux. Les classes d'événements à collecter depuis le réseau sont sélectionnées depuis ces schémas d'attaques, plus précisément depuis les *filtres d'événement*.

La *fonction d'interactio*n permet aux agents de se communiquer leurs analyses et leurs décisions. Elle permet aussi à l'administrateur d'interagir avec l'agent gestionnaire de politiques de sécurité.

Grâce à la fonction de *délibération*, l'agent est capable de raisonner et d'extrapoler en s'appuyant sur ses aptitudes mentales (croyances, buts et suspicions), sa connaissance intégrée et son expérience, d'une manière rationnelle, pour trouver les réponses appropriées. Quand un événement survint sur le réseau, un *agent surveillant local* teste si il correspond à la classe de l'événement spécifié. Si il y a correspondance, il est collecté. Puis, *les croyances des agents* sont créées et mises à jour depuis l'événement collecté. Ces croyances sont ensuite analysées pour tester si elles correspondent à *un schéma d'attaque*. Si il y a correspondance, *un but de détection* est atteint. Les *croyances* et *suspicions* des autres agents sont aussi considérées quand un agent analyse ses *croyances*. Quand un but est atteint (i.e. une attaque est détectée), une liste d'actions (comme informer l'administrateur) est exécutée.

Dans AMASIR, les agents ne sont pas capable de détecter des attaques nouvelles et inconnues. C'est pourquoi notre objectif est d'étendre le modèle existant avec un algorithme de Case-Based Reasoning pour :

- obtenir de nouveaux événements à analyser,
- détecter des attaques inconnues, grâce à ces événements,
- créer, à partir de ces nouvelles attaques détectées, de nouveaux schémas d'attaque,
- mettre à jour les politiques de sécurité avec ces nouveaux schémas d'attaques à détecter et créer de nouveaux buts de détection à envoyer aux agents appropriés.

Dans la prochaine section, nous allons présenter notre approche d'apprentissage.

3. Notre approche d'apprentissage

Dans les systèmes de détection d'intrusion existants, la notion d'apprentissage est utilisée pour apprendre les profils normaux des utilisateurs et des systèmes à sécuriser. Ces profils normaux sont ensuite comparés avec les profils courants pour détecter les attaques potentielles. Dans notre cas, nous proposons d'utiliser la notion d'apprentissage pour apprendre les profils anormaux qui correspondent aux attaques.

Dans [1], nous avons souligné l'importance, pour un système de détection d'intrusion, d'apprendre de nouvelles structures d'attaques afin de détecter les nouvelles attaques quand elles se produisent. Une des propriétés principales des AI est leur capacité d'apprentissage. Ainsi, dans ce papier, nous nous focalisons sur cette propriété en ajoutant une fonction d'apprentissage dans l'architecture interne des agents d'AMASIR. Pour ajouter cette fonction, les agents vont utiliser les schémas d'attaque existants et les attaques ayant survenues dans le passé pour identifier des similarités avec une séquence d'événement courante suspicieuse, qui ne correspond ni à une séquence normale, ni a un schéma d'attaque (i.e. une attaque connue).

Quand des similarités sont identifiées, l'administrateur va être notifié et ainsi être capable de confirmer si une attaque survient ou pas. Si une attaque est confirmée, les différents agents vont intégrer le nouveau schéma d'attaque à leur base de donnée. Cela peut être réalisé de deux manières différentes :

- l'administrateur spécifie le nouveau schéma d'attaque à *l'agent gestionnaire de politique de sécurité*.
- *l'agent gestionnaire* construit le nouveau schéma d'attaque depuis les événements observés par ses *agents surveillants locaux*.

Ainsi, durant la phase d'apprentissage, les agents d'AMASIR vont interagir de la manière suivante :

- les agents surveillants locaux vont coopérer pour identifier une suite suspicieuse d'événement,
- les agents surveillants locaux vont communiquer avec leur agent gestionnaire qui lui-même va interagir avec l'administrateur pour confirmer la détection d'une attaque
- l'administrateur va interagir avec l'agent gestionnaire de politique de sécurité pour apprendre de nouveaux schémas d'attaque.

De plus, comme nous l'avons dit précédemment, nous proposons d'ajouter un moteur de Case-Based Reasoning aux agents surveillants locaux pour permettre aux agents d'identifier des similarités entre attaques passés et présentes.

Dans la prochaine section, nous allons présenter rapidement cette technique de Case-Based Reasoning.

4. La technique de Case-Based Reasoning

Comme le définissent Aamodt & Plaza ([9]), "*Le Case-Based Reasoning (CBR), est une approche capable d'utiliser la connaissance spécifique de problèmes (cas) s'étant déjà produits dans le passé pour répondre à de nouveaux problèmes. Un nouveau problème est résolu en cherchant un cas passé similaire, et en réutilisant*

sa solution dans le problème présent." Cette définition montre que le CBR est un algorithme très naturel : en tant qu'être humain, lorsqu'un problème se pose, l'idée première est de rechercher si ce problème (ou un cas similaire) s'est déjà produit, et, le cas échéant, reprendre la solution passée pour essayer de l'adapter au problème présent.

Les principales étapes d'un algorithme CBR sont [9] :

- le "*Retrieve*". La première étape est de chercher, dans la base de donnée des problèmes résolus, le cas le plus similaire au problème présent.
- le "*Reuse*". La seconde étape est de réutiliser la solution passée pour le problème courant.
- le "*Revise*". La troisième étape est d'adapter la solution passée au problème courant, afin d'obtenir une nouvelle solution.
- le "*Retain*". L'étape finale consiste à stocker la solution adaptée dans la base de donnée des cas.

Plusieurs systèmes de CBR ont été développés dans des domaines variés.

PROTOS est un système de CBR développé pour aider au diagnostique des malentendants [10], [12]. Dans ce système, un problème est présenté comme un vecteur de caractéristiques, et la tâche du CBR est de retrouver le cas passé qui correspond le plus au vecteur. Une solution à un problème passé est proposée comme solution au problème présent sans adaptation (pas de Revise). Si la solution est refusée par l'utilisateur, une étape d'apprentissage commence avec l'aide de l'utilisateur qui peut proposer une nouvelle solution ou alors demander la recherche d'une autre solution.

CASEY est un système de CBR développé pour aider au diagnostique de problèmes cardiaques [10], [13]. Un problème présenté à ce système est résolu en recherchant un cas, et, contrairement à PROTOS, en adaptant la solution passée au problème présent. Le problème majeur de CASEY est son absence d'interaction avec l'utilisateur.

CHROMA est un système fondé sur des agents et utilisé pour choisir les techniques de chromatographie utilisées dans la purification de protéines issues de cultures [13]. Ce système se focalise plus précisément sur deux modes de coopération dans un CBR distribué, entre des agents disposant de connaissances propres : le CBR distribué (DistCBR) et le CBR collectif (ColCBR). Dans le premier mode, le problème posé à un agent A est transmit par celui-ci à un autre agent B. Ainsi, la méthode de CBR utilisée sera celle de B. Dans le second mode lorsque A transmet le problème, il transmet également sa méthode à B (ainsi, la méthode de A sera utilisée avec la connaissance de B). Autrement dit, l'émetteur utilise les mémoires de cas des autres agents : il s'agit, en quelque sorte, d'une mémoire collective.

Il existe de nombreux documents traitant des optimisations de l'algorithme et plus particulièrement :

- de l'organisation des données dans la base de cas,
- de l'algorithme de similarité utilisé pour la recherche du meilleur cas,
- de l'utilisation de connaissances pour effectuer une analyse sémantique lors du *Retrieve* et du *Retain* au lieu de l'analyse syntaxique traditionnellement utilisée.

Dans la section suivante, nous allons proposer un algorithme de CBR qui sera intégré aux différents agents d'AMASIR (à l'exception de *l'agent gestionnaire de politique de sécurité*).

5. Un modèle d'apprentissage pour AMASIR

Pour apprendre de nouvelles attaques, nous proposons d'intégrer aux différents agents d'AMASIR (à l'exception de *l'agent gestionnaire de politique de sécurité*), une fonction d'apprentissage fondée sur un modèle de CBR (la fig. 2 montre le modèle d'apprentissage).

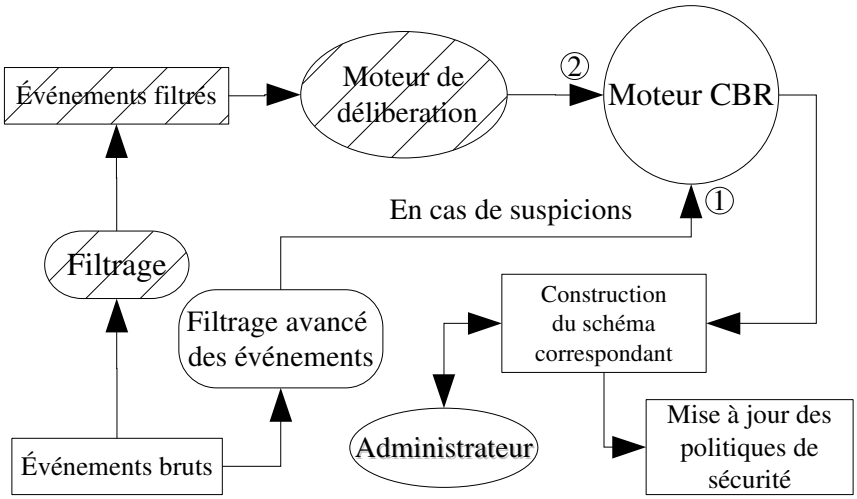


Fig 2: Vue interne du modèle d'apprentissage

Pour ajouter la fonction d'apprentissage, nous avons défini un modèle d'apprentissage constitué de trois sous-modèles [15] :

- un modèle de filtrage avancé des événements, qui permettra de filtrer des nouveaux événements en étendant la fonction existante de filtrage d'événement,

- un modèle de Case-Based Reasoning, qui reconnaîtra de nouvelles attaques en utilisant un moteur de CBR,
- un modèle de génération de schéma d'attaque, qui construira automatiquement ou semi-automatiquement un nouveau schéma d'attaque à partir des caractéristiques de l'attaque nouvellement détectée. Ce modèle permettra alors de mettre à jour les politiques de sécurité afin de prendre en compte les nouvelles attaques.

5.1. *Le modèle de filtrage avancé des événements*

Comme décrit précédemment dans la section 2, chaque agent local possède un filtre d'événement qui sélectionne les événements à collecter suivant les caractéristiques des attaques qu'il a à détecter (i.e. ses buts). Ce filtrage n'est pas suffisant puisque il réduirait le nombre de nouvelles attaques pouvant être détectées. C'est pourquoi nous proposons d'ajouter un autre filtre qui permettra de collecter de nouvelles sortes d'événements qui ne seront pas analysés par le moteur de délibération mais directement par le moteur CBR.

5.2. *Le modèle de CBR*

Comme expliqué précédemment, le but du modèle de CBR est d'identifier de nouvelles attaques comme étant des variations d'attaques connues. Cette identification est effectuée en trois étapes (la fig. 3 présente le moteur de CBR).

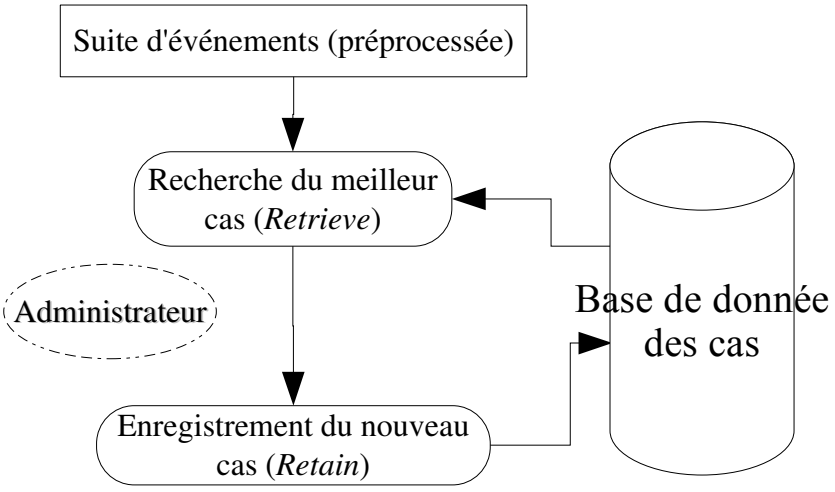


Fig 3: Le moteur de CBR

Quand une séquence d'événement est filtrée, elle est traitée par un préprocesseur qui agrège les événements identiques contenus dans cette séquence. Puis, en utilisant une fonction de similarité, le cas le plus similaire est recherché dans une base de donnée des cas, qui contient des attaques connues (i.e. des références/schémas d'attaques) et les séquences d'événement précédentes identifiées comme attaques par le moteur de CBR. A cette étape, si le meilleur taux de similarité n'est pas assez élevé, le moteur de CBR devra interagir avec l'administrateur pour défausser la suite d'événement ou confirmer qu'il s'agit d'une nouvelle attaque. Si la séquence d'événement est identifiée comme une attaque, elle est ajoutée dans la base de donnée des cas. Puis, le générateur de schéma d'attaque crée un nouveau schéma d'attaque. Ce schéma sera ensuite ajouté à la base de donnée des schémas d'AMASIR.

5.2.1. *La fonction de similarité*

L'objectif de cette fonction de similarité est de trouver durant une phase de *Retrieve* le cas le plus similaire, parmi une base de donnée de cas, à une séquence d'événement suspicieuse. Au préalable, la séquence suspicieuse est mis par un préprocesseur sous une forme contenant :

- des opérateurs d'occurrence permettant d'agréger plusieurs éléments identiques consécutifs,
- des opérateurs de séquence permettant de définir si l'ordre temporel des événements est caractéristique de l'attaque.

Avant de définir la fonction de similarité, regardons la hiérarchie des classes d'événement définis dans AMASIR (cette hiérarchie est présentée fig. 1). Au niveau 0, les événements sont vus comme des événements génériques (GenericSecurityEvent). Le niveau 1 représente les événements d'audits (i.e. collectés de journaux systèmes) et les événements réseaux. Finalement, le niveau 2 représente des événements spécifiques, tel que des événements TCP, UDP ou ICMP.

Dans les sections suivantes, nous discuterons sur la manière dont la fonction de similarité est appliquée à trois niveaux pour comparer les différents attributs des événements constituant la séquence à analyser. En effet, nous allons décrire les algorithmes permettant de : 1) comparer deux événements, 2) comparer deux suites d'événement et 3) chercher le cas le plus similaire, pour une séquence suspicieuse donnée, parmi une base de donnée de séquences de référence.

5.2.1.1. Principes de base de la fonction de similarité

Notre fonction se fonde sur l'algorithme le plus simple existant dans le domaine du Case-Based Reasoning : CBL1 [11][16]. Nous modifions cet algorithme en lui ajoutant un coefficient de pondération défini comme :

$$\sum_{i \in P} w_i = 1, \quad 0 < w_i \leq 1 \tag{1}$$

A présent, nous pouvons définir une fonction fondamentale pour comparer deux événements au niveau 0 :

$$Similarity(C_1, C_2, P, w) = \frac{1}{\sqrt{\sum_{i \in P} w_i \cdot Feature\_dissimilarity(C_{1_i}, C_{2_i})}} \tag{2}$$

Avec :

$$Feature\_dissimilarity(C_{1_i}, C_{2_i}) = \begin{cases} (C_{1_i} - C_{2_i})^2 & si\ la\ caractéristique\ i\ est\ numérique \\ 0 & si\ la\ caractéristique\ i\ est\ discrète\ et\ C_{1_i} = C_{2_i} \\ 1 & sinon \end{cases} \tag{3}$$

Le problème avec une telle fonction est que les caractéristiques *discrètes* peuvent avoir plus de poids que les caractéristiques *non discrètes*. Pour cette raison, nous considérons, en créant P, que tous les attributs d'événements sont *discrets*. Le choix du *vecteur de pondération* w est un autre problème. Une solution possible est de créer ce vecteur suivant les caractéristiques que nous souhaitons privilégier. Notons que, pour ne pas surcharger ce papier, nous supposerons que toutes les caractéristiques ont le même ordre de grandeur. En fait, nous devrions effectuer une *normalisation* pour nous passer de cette hypothèse : cette *normalisation* est décrite dans [15].

5.2.1.2 Comparaison en profondeur de deux événements

A présent, concentrons nous sur la comparaison de deux événements en prenant en compte tous les niveaux de la hiérarchie des classes d'événement d'AMASIR :

$$Similarity_{AV}(C_1, C_2) = Moyenne_H(Similarity(C_1, C_2, P), Similarity_2(C_1, C_2, P')) \tag{4}$$

Cette comparaison avancée est effectuée en calculant la moyenne harmonique ($Moyenne_H$) des comparaisons effectuées au niveau 1 et au niveau 2. L'utilisation de

cette sorte de moyenne se justifie par le fait que nous travaillons avec des *inverses de distances*, et qu'une moyenne standard nous semble dans ce cas là, *intuitivement*, moins pertinente qu'une moyenne harmonique (qui est *un inverse de moyenne d'inverse de distances*). Les travaux futurs comporteront une preuve formelle de cette intuition...

La fonction $Similarity_2$ permet d'effectuer la comparaison au niveau 1 et 2. Cette fonction est détaillée dans [15].

5.2.1.3 Comparaison de deux séquences d'événement

Après avoir défini la fonction permettant de comparer deux événements, comment comparer deux séquences d'événement ? Nous définissons pour cela la fonction $Comparaison_S$ (S pour Séquence) :

$$Comparaison_S(C_1, C_2) = \begin{cases} Comparaison_{Atemp}(C_1, C_2) & Si\ l'ordre\ temporel\ n'est\ pas\ significatif\ dans\ C_2 \\ Comparaison_{Temp}(C_1, C_2) & Si\ l'ordre\ temporel\ est\ significatif\ dans\ C_2 \end{cases} \quad (5)$$

Cette fonction effectue deux comparaisons différentes suivant que l'ordre temporel est significatif ou non dans le cas de référence.

Cas où l'ordre temporel est non significatif (fonction A_{Atemp}):

Considérons : 1) **A**, une séquence suspicieuse d'événement, avec $\mathbf{A_i}$ représentant les différents événements de cette séquence et 2) **B**, une base de donnée de cas de référence, avec $\mathbf{B_j}$ un cas de référence et $\mathbf{(B_j)_k}$ un événement de ce cas de référence.

$$Comparaison_{Atemp}(A, B_j) = \begin{cases} Pour\ chaque\ (B_j)_k\ (événement) : \\ \quad Pour\ chaque\ A_i\ (événement) : \\ \quad\quad Calculer\ S_{(i\ k)} = Similary_{AV}(A_i, (B_j)_k) \\ Pour\ chaque\ (B_j)_k\ (événement) : \\ \quad Calculer\ SMAX_k = DS(S, k) \\ Calculer\ S_{Total} = Moyenne_H(SMAX_k) \end{cases}$$

$$avec\ DS(S, k_n) = \begin{cases} Chercher\ i_{max}\ tel\ que\ S_{(i_{max}\ k_n)} = Max(S_{(i\ k)}),\ pour\ k \in [1, N_{(B_j)}] \\ si\ i_{max}\ n'est\ pas\ marqué \\ \quad si\ (S_{(i_{max}\ k_n)}) = Max(S_{(i\ k)}),\ pour\ i \in [1, N_A],\ i\ non\ marqué \\ \quad\quad marque\ i\ et\ renvoie\ S_{(i_{max}\ k_n)} \\ sinon\ recommencer\ en\ cherchant\ i_{(max-\ 1)},\ etc... \end{cases} \quad (6)$$

Le but de la fonction DS (*Distribute Similarities)* est de trouver le meilleur accord entre les événements $\mathbf{A_i}$ de la séquence suspicieuse **A** et les événements $\mathbf{(B_j)_k}$ du cas de référence $\mathbf{B_j}$ sans prendre en compte l'ordre temporel. Cette fonction est décrite plus en détail dans [15].

Cas où l'ordre temporel est significatif (fonction A_{Temp}):

Comme précédemment, considérons les même éléments **A** et **B**. L'idée de l'algorithme est de prendre chaque événement $\mathbf{(B_j)_k}$ du cas de référence $\mathbf{B_j}$ et de trouver avec quel événement $\mathbf{A_i}$ de la suite suspicieuse **A** il est le plus similaire (i.e. c'est une recherche de la similarité maximale). Quand la position de l'événement le mieux "assorti" est connue, nous stockons son index (i.e. le couple (i_{max}, k)) et nous créeons un sous-ensemble **V** qui contient les événements de **A**, commencant par l'événement suivant l'index stocké et finissant à la fin de la séquence. Ainsi, dans l'étape suivante, la recherche de l'événement le mieux "assorti" pour l'événement suivant de $\mathbf{B_j}$ sera effectué dans **V** seulement. Par conséquent, pour chaque événement $\mathbf{(B_j)_k}$ de $\mathbf{B_j}$, le vecteur **V** crée à partir de **A** se réduira au fur et à mesure. Ainsi, le fait de rechercher l'événement suivant $\mathbf{(B_j)_{k+1}}$ dans un sous-ensemble de **A** permet de guarantir l'ordre temporel (et d'ignorer les événements de **A** qui sont redondants ou non contenus dans $\mathbf{B_j}$).

$$Comparaison_{Temp}(A, B_j) = \begin{cases} première\_comparaison() \\ \\ Pour\ chaque\ (B_j)_k\ (événement)\ : \\ \quad Soit\ V\ un\ sous\ intervalle\ de\ [(i_{max}+1, k-1), N_A] \\ \quad Si\ V=\emptyset \\ \quad\quad S_{(i_{max}k)}=1 \\ \quad Sinon \\ \quad\quad Pour\ chaque\ A_i\ (événement),\ i\in V \\ \quad\quad\quad Calculer\ S_{(i\ k)} = Similarity_{AV}(A_i, (B_j)_k) \\ \quad\quad Chercher(i_{max}, k)\ tel\ que\ S_{(i_{max}\ k)} = Max(S_{i\ k})\ pour\ i\in V \\ \quad\quad Si\ S_{(i_{max}\ k)} < T_{min} \\ \quad\quad\quad (i_{max}, k)=(i_{max}, k-1)\ et\ S_{(i_{max}\ k)}=1 \\ Calculer\ S_{Total} = Moyenne_H(S_{(i_{max}\ k)}), pour(k\in[1, N_{(B_j)}]) \end{cases} \tag{7}$$

$$avec\ première\_comparaison = \begin{cases} Pour\ (B_j)_1, \\ \quad Pour\ chaque\ A_i\ (événement)\ : \\ \quad\quad Calculer\ S_{(i\ 1)} = Similarity_{AV}(A_i, (B_j)_1) \\ Chercher(i_{max}, 1)\ tel\ que\ S_{(i_{max}\ 1)} = Max(S_{i\ 1})\ pour\ i\in[1, N_A] \\ Si\ S_{(i_{max}\ 1)} < T_{min} \\ \quad (i_{max}, 1)=(1,1)\ et\ S_{(i_{max}\ 1)}=1 \end{cases}$$

Un problème peut survenir si un événement $\mathbf{(B_j)_k}$ du cas de référence $\mathbf{B_j}$ n'existe pas dans la suite suspicieuse **A**. En fait, dans ce cas, toutes les valeurs des

similarités pour cet événement seraient médiocres et **V** pourrait se réduire considérablement en "écrasant" des événements importants. Pour cette raison, nous utilisons une constante $\mathbf{T_{min}}$ qui garantit une valeur de similarité minimale avant de réduire **V**. Si, à un moment donné, un événement de $\mathbf{B_j}$ n'existe pas dans **A**, la valeur de similarité maximale pour cet événement serait médiocre. Nous la remplacerions donc par 1 (qui signifie aucune similarité) et continuerions avec l'événement suivant sans réduire **V**.

5.2.1.4 Comparaison finale

Nous pouvons à présent définir la fonction de comparaison finale pour rechercher le cas $\mathbf{B_{best}}$ le plus similaire à une séquence suspicieuse **A** dans une base de donnée de cas de référence **B**.

$$Comparaison_{Finale}(C, B) = \begin{cases} Pour\ chaque\ B_j,\ calculer\ S_j = Comparaison_S(C, B_j) \\ Le\ meilleur\ cas\ est\ B_{Best},\ tel\ que\ S_{Best} = Max(S_j),\ j \in [1, N_B] \end{cases} \quad (8)$$

5.2.2. *Optimisation*

Intéressons nous à présent à la complexité des trois fonctions, avec n le nombre d'événements dans la suite suspicieuse, m le nombre moyen d'événements dans un cas de référence et M le nombre de cas de référence :

Intéressons nous à présent à la complexité des trois fonctions, avec **n** le nombre d'événements dans la suite suspicieuse, **m** le nombre moyen d'événements dans un cas de référence et **M** le nombre de cas de référence :

- $Comparaison_{Atemp}$: environ o(2.**m**.**n**).
- $Comparaison_{Temp}$: environ o(2.**m**.**n**) au pire des cas (i.e. *V* ne diminue que d'un élément à la fois).
- $Comparaison_{Finale}$: environ o(2.**m**.**n**.**M**) + o(**M**) (pour le calcul du maximum).
- Voici un ordre de grandeur des entités utilisées par la fonction de similarité :
- Nombre d'événements de A (i.e. N_A) : 10.
- Nombre moyen d'événements dans une attaque de référence : 10.
- Nombre de cas de référence dans la base : 1000.

Nous nous apercevons que la complexité du système augmente quadratiquement quand la taille des données augmente linéairement, ce qui n'est pas satisfaisant.

De nombreuses améliorations ont été proposées pour CBL1, portant sur des éléments divers tels que l'organisation des structures de données de la base des cas, le pré-traitement des données, etc...[9][11]. Sans rentrer dans des améliorations complexes, nous proposons de diminuer le nombre de calculs de similarité inutiles grâce à cet algorithme : Avant qu'une suite suspicieuse soit analysée, nous construisons une table d'index contenant les noms des événements (attribut *EventName* de la classe *GenericSecurityEvent*) ainsi que leur numéro (voir fig. 4).

Par exemple, la séquence suivante :

$$\begin{bmatrix} Event\ 1 & EventName = Telnet \\ Event\ 2 & EventName = SSH \\ Event\ 3 & EventName = Telnet \end{bmatrix}$$ *a la table d'index suivante :* $$\begin{bmatrix} EventName = Telnet & [1,3] \\ EventName = SSH & [2] \end{bmatrix}$$

Fig 4: Construction d'une table d'index

Ainsi, au lieu de parcourir tous les événements $\mathbf{A_i}$ d'une suite **A** (par exemple dans $Comparison_{Temporal\_sequence}$), pour calculer leur similarité avec un événement $\mathbf{(B_j)_k}$, nous effectuons des calculs de similarité uniquement entre des événements ayant la même valeur de l'attribut *EventName*. Cela revient à faire une approximation, puisque nous ne testons que les cas susceptibles d'offrir une similarité élevée, mais cela nous permet de diminuer la complexité du système : en considérant que nous ne balayons plus que 10% des événements de **A**, et en utilisant l'ordre de valeur énoncé plus haut, la complexité des fonctions devient :

- $Comparaison_{Atemp}$: ~ o(2.**m**) + o(**n**).
- $Comparaison_{Temp}$: ~ o(2.**m**) + o(**n**).
- $Comparaison_{Finale}$: ~ o(2.**m**.**M**) + o(**M**) + o(**n**)

Note : les o(**n**) dans chaque comparaison proviennent de la construction de la table d'index, qui se fait en temps linéaire.

La complexité de l'algorithme reste quadratique, mais dans la comparaison de deux suites devient linéaire. Une mise en pratique de ces tables d'index à un niveau plus haut (i.e. pour comparer la suite suspicieuse avec seulement les cas intéressants) serait intéressante pour obtenir une complexité logarithmique sur tout l'algorithme. Cette optimisation, ainsi que toutes les autres, ouvre la possibilité à de nombreux travaux futurs.

5.3. *Le modèle de génération de schéma d'attaque*

Le but du **modèle de génération de schéma d'attaque** est de mettre à jour la base de schémas d'attaque. La phase de mise à jour commence quand une séquence

d'événements suspicieuse est identifiée comme étant une nouvelle attaque. A ce moment là, il est nécessaire de générer automatiquement le schéma d'attaque lié à la séquence d'événement suspicieuse. Pour cela, le générateur de schéma d'attaque doit déterminer quels événements sont caractéristiques de l'attaque, quels attributs (par exemple la source de l'attaque) peuvent être omis, si le temps entre deux événements est significatif, si l'ordre temporel est nécessaire, etc... Quand un schéma d'attaque est crée, et après confirmation de l'administrateur, il est enregistré dans la base des schémas d'attaque. Nous considérons que la génération du schéma d'attaque est un point trop critique pour être effectué automatiquement. Ainsi, une intervention humaine est nécessaire avant de mettre à jour la base de schéma d'attaque.

6. Implémentation

Nous avons commencé à implémenter notre modèle de CBR en Java, avec la librairie *Jpcap* nous permettant de lire des fichiers au format *pcap*. Cependant, nous n'avons pu finir nos tests cas le préprocessing et la phase de "*Retrieve*" étaient trop coûteux en terme de ressources. C'est pourquoi nous sommes en train de travailler sur une implémentation en C de notre modèle. En ce qui concerne la base de donnée des cas, elle a été initialement remplie avec le jeu de données DARPA Lincoln Laboratory off-line evaluation data set 1999 (IDEVAL) [17], [18].

7. Conclusion

Dans ce papier, nous avons décrit notre modèle d'apprentissage fondé sur une approche CBR pour détecter et apprendre des attaques inconnues. Pour apprendre une nouvelle attaque, nous avons dû tout d'abord la détecter, puis mettre à jour la base de donnée des schémas d'attaque. La caractéristique principale de notre modèle d'apprentissage est qu'il ne nécessite pas la construction d'un profil normal avant la phase de détection. Ce modèle d'apprentissage a été intégré dans un système de détection d'intrusion existant nommé AMASIR, qui visait à détecter uniquement des attaques connues. Ainsi, les différents agents d'AMASIR (à l'exception de *l'agent gestionnaire de politiques de sécurité*) ont été améliorés avec une fonction d'apprentissage. A présent, nous travaillons sur l'implémentation de cette fonction et plus précisément sur le moteur de CBR. Dans des travaux futurs, nous prévoyons de continuer les spécifications du modèle de génération de schéma d'attaque.

8. Bibliographie

1. Boudaoud, K.: Intrusion Detection: a New Approach using a Multi-agents System. PhD thesis, Institut Eurecom/EPFL, Sophia Antipolis (2000) .

2. Boudaoud, K., McCathieNevile, C.: An Intelligent Agent-based Model for Security Management. Proceedings of the Seventh IEEE Symposium on Computers and Communications (ISC'02), Taormina, Italy (July 2002).

3. Lee, W., Stolfo, S. J., Mok, K. W.: A Data Mining Framework for Building Intrusion Detection Models. Proceedings of the IEEE Symposium on Security and Privacy (1999).

4. Helmer, G., Wong, J. S. K. Honovar, V., Miller, L.: Automated Discovery of Concise Predictive Rules for Intrusion Detection. Journal of Systems and Software 60 (2002).

5. Lee, W., Stolfo, S. J.: Data Mining Approaches for Intrusion Detection. Proceedings of the 7th USENIX Security Symposium (1998).

6. Stolfo, S., Prodromidis, A. L., Tselepis, S., Lee, W., Fan, D. W., Chan, P. K.: Java Agents for Meta-Learning over Distributed Databases. American Association for Artificial Intelligence (1997).

7. Barbará, D., Couto, J., Jajodia, S., Popyack, L., Wu, N.: ADAM : Detecting Intrusions by Data Mining. Proceedings of the 2001 IEEE Workshop on Information Assurance and Security (2001).

8. Portnoy, L.: Intrusion Detection with Unlabeled Data Using Clustering. ACM Workshop on Data Mining Applied to Security (DMSA' 2001).

9. Aamodt, A., Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications Vol. 7 Nr. 1 (1994).

10. Aamodt, A.: Knowledge-Intensive Case-Based Reasoning and Sustained Learning. Proceedings of the 9th European Conference on Artificial Intelligence (ECAI-1990).

11. Aha, D. W.: Case-Based Learning Algorithms. Proceeding of the 1991 DARPA Case-Based Reasonning Workshop (1991).

12. Porter, B.: PROTOS; An Experiment in Knowledge Acquisition for Heuristic Classification Tasks. Proceedings of the First Intern. Meeting on Advances in Learning. Les Arcs, France (1986) 159-174.

13. Koton, P.: Reasoning about Evidence in Causal Explanations. Proceedings of AAAI-88 (1988) 256-261.

14. Plaza, E., Arcos, J. L., Martin, F.: Cooperation Modes among Case-Based Reasoning Agents. Proceeding of the ECAI'96 Workshop on Learning in Distributed AI Systems (1996).

15. Nobelis, N.: Master Thesis, University of Nice Sophia Antipolis, France (2004).

16. Wettschereck, D., Aha, D. W.: Weighting Features. ICCBR-95 (1995).

17. Mahoney, M. V., Chan, P. K.: An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection. In: Recent Advances in Intrusion Detection (RAID2003). Lecture Notes in Computer Science, Vol. 2820. Springer-Verlag, (2003) 220-237

18. Haines, J. W., Lippmann, R. P., Fried, D. J., Tran, E. Boswell, S., Zissman, M. A.: DARPA Intrusion Detection System Evaluation: Design and Procedures. Technical Report (2001).

Matrix approach for anomalies detection and correction in firewall filtering rules

Mohammed Anis BENELBAHRI

College of Telecommunications (Sup'Com),
Cité Technologique des Communications
Route de Raoued Km 3,5 – 2083 El Ghazala, Ariana, Tunisia
benelbahri@yahoo.fr

Adel BOUHOULA, Zouheir TRABELSI

College of Telecommunications (Sup'Com),
Route de Raoued Km 3,5 – 2083 El Ghazala, Ariana, Tunisia
bouhoula@planet.tn, trabelsi.zouheir@supcom.rnu.tn

ABSTRACT. *Firewalls are basic components in the network security architecture. They filter the network traffic based on a set of predefined filtering rules. Hence, the filtering rules have to be well defined and coherent in order to guarantee the desired responses of the Firewall. In this paper, we propose an approach for detecting anomalies in the Firewall filtering rules. The list of the filtering rules is represented by a matrix. Then, we generate a number of matrices defining all the relationships between rules. Each matrix is related to a type of network packet's field. Finally, from these matrices, we easily detect the anomalies within filtering rules. In addition, we propose a set of actions to correct the anomalies, namely the relaxed actions. Relaxation means that we tolerate the existence of some anomalies within filtering rules. In addition, the approach works with any filtering rule's format. Also, we have implemented it and the experimental result shows its simplicity and its efficiency.*

RÉSUMÉ. *Les firewalls sont des composants de base dans la sécurité des réseaux. Ils filtrent le trafic réseau en se basant sur une liste prédéfinie de règles de filtrage. En outre, les règles de filtrage doivent être cohérentes et bien définies afin d'assurer la réponse désirée du firewall. Dans cet article, nous proposons une approche pour la détection des anomalies dans les règles de filtrage. Une liste de règles de filtrage est représentée à l'aide d'une matrice à partir de laquelle il est possible de générer un nombre de matrices définissant les relations entre les règles. Chaque matrice concerne un champ dans la règle. Enfin, et à partir de ces matrices, il est facile de détecter la présence des anomalies. Ensuite, nous proposons un ensemble d'actions relaxées pour corriger les anomalies. En effet, la relaxation consiste à tolérer certaines anomalies. Notre approche est indépendante du format des règles de filtrage, nous l'avons implémentée et les premières expérimentations montrent sa simplicité et son efficacité.*

KEYWORDS: *Firewalls, Filtering rules, Anomalies, Security Policy, Security Policy Conflicts.*
MOTS-CLÉS : *Firewalls, Règles de filtrage, Anomalies, Politique de Sécurité, Conflits.*

1. Introduction

Security policy in a firewall is generally described by a list of filtering rules. Generating these rules is considered a heavy task, due to the network complexity (many network segments), the great number of network's equipments (personal computer, servers, routers, etc.), and the number of vulnerabilities within routers, servers, etc. Weak or ill defined filtering rules can modify the expected and desired responses of the Firewall and consequently increase the number of attacks on the network.

Moreover, the manual generation of the filtering rules is, in general, followed by a number of faults which are transformed into anomalies altering the normal operation of the filtering process. In this context, we propose, in this paper, an approach for detecting and correcting the anomalies within the filtering rules. By generating filtering rules free from anomalies, the Firewall can filter any network packet and produce the desired and expected filtering results.

Several approaches were proposed in order to define the existing anomalies in the firewall. (Hari *et al*., 2000) consider that a filter conflict occurs when two or more filters overlap and create an ambiguity in packet classification. The authors propose a scheme for conflict resolution, which is based on the idea of adding resolve filters. They formally characterize the conditions which lead to conflicts amongst filters. (Hamed *et al*., 2003*)* present, in their paper a set of techniques and algorithms that provide firstly automatic discovery of firewall policy anomalies to reveal rule conflicts and potential problems in legacy firewalls, and secondly anomaly-free policy editing for rule insertion, removal and modification. (Razvani *et al*.) present a language for high level and formal specification of security policy in firewalls. The language is based on the deontic logic. It supports separation of the security policy from the network topology and an automatic generation of an existing firewall rules. According to (Razvani *et al*.), analyzing and verifying security policy include proving consistency in security policy, proving coverage among security propositions and applying a query on the policy.

On the other hand, many methods are applied to represent and analyse rules. For example, ordered binary decision diagrams (BDDs) discussed in (Zaliva *et al*.) are a potential method of representing the rules. This paper presents an algorithm for representing the filtering lists as a BDD and then shows how the resulting Boolean expression can be used to analyze rule sets.

Similar work based on a logic background has been done by (Hazelhurst *et al*.). They have used ordered binary decision diagrams to analyze access lists. This representation allows efficient manipulation of the lists, and finding redundant rules is easy, for instance. However, the system does not allow expressing custom rules using a logic programming syntax. Several researches have also implemented tools for describing the contents of an access list based on other approaches. Guttman describes an approach for generating filters based on a security policy and verifying

that a packet filter implements some security policy (Guttman, 1997). (Hari *et al.* 2000) have applied these techniques to analyzing access lists from a security viewpoint.

Many other tools were developed to analyze firewall configurations and they usually rely on hard-coded algorithms for analyzing access lists. (Eronen *et al.*) presents a tool based on constraint logic programming (CLP) which allows the user to write higher level operations for, e.g., detecting common configuration mistakes.

However, the proposed approach assumes that a network packet is a set of fields. For each field, we define a matrix specifying the relationship between each two filtering rules regarding that particular field. Then, all the generated matrices are used to detect anomalies within the filtering rules. Therefore, any filtering rule format can be treated. A correction model of the filtering rules anomalies is then proposed. We will discuss the suggested correction's actions for each anomaly.

Moreover, we will address two important issues related to the phase of the post correction process, namely the issue of reordering filtering rules and inserting new rules. Finally, we present a concrete example of a set of filtering rules and we generate the related matrices. The resulting matrix will show clearly the filtering rules that have anomalies and their types.

2. Matrix representation

The process of detecting and correcting anomalies in Firewall filtering rules is a crucial step in ensuring that the Firewall would apply the correct filters on the network packets. In addition, such a process would improve the Firewall performance.

Each network packet is a set of fields. For each particular field, we define the relationship between each two filtering rules by a matrix. The matrices generated would be used to detect anomalies in the filtering rules.

2.1. Definitions: Packet's fields and filtering rules

The following are the definitions related to packet's fields and filtering rules.

Definition 1 [Field]

A field is the atomic element. Its value can be a single (or a finite set of) value(s), named a domain.

Definition 2 [Filtering rule, Filter]

- A filtering rule is a finite set of fields.
- A filter is a finite list of filtering rules.

2.2. Filtering rule format

A filtering rule is the concatenation of header fields and the action field. Its general format is the following: ***<Header> <Action>***

Headers format depends on the protocol type. The following are examples:

1. UDP Header

<Protocol > < source IP > < dest IP > < source Port > < dest Port >

2. TCP Header

<Protocol > < source IP > < dest IP > < source Port > < dest Port > <ACK bit>

3. ICMP Header

<Protocol> < source IP > < destination IP > <Type> <Code>

4. ARP/RARP Header

<Protocol> < source IP > < destination IP >

2.3 The header format

Let *H* be the set of fields of a given header format. The dimension of H is the number of fields in the header. So H is defined as the following:

$$H=\{h_1,h_2,...,h_m\}, \dim(H)=m.$$

We define also $val(h_j)$ as the value of the field h_j. $val(h_j)$ can be a single or a set of value(s).

2.4 Filter matrix

In the following, we denote a filter by a matrix; called *F*. The filtering rules will be the rows of the matrix F. And, the set of values of the fields that have the same type are the vectors of F.

Figure 1 shows the algebraic writing of the matrix F where *n* is the number of rows (or the number of filtering rules) and *m* is the number of columns (or field's number).

$$F=(a_{ij})_{\substack{1\le i\le n\\ 1\le j\le m}}=\begin{pmatrix} a_{11} & . & . & . & . & a_{1m} \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ a_{n1} & . & . & . & . & a_{nm} \end{pmatrix}$$

Figure 1. *Matrix F*

2.5 Fields relationships and rules relationships

Referring to the previous definitions, the second observation is focused on the relations between the filtering rules. In fact, we can notify that the relationships between the filtering rules can be deduced from the relationships between their different fields. In the following, we present the definitions of the different relationships between two given filtering rules R_i and R_j.

The two filtering rules R_i and R_j are said to be equal iff all of their fields are equal.

Definition 3 [Equality Relationship]

$$R_i = R_j \ \ iff \ \ \forall k = 1..m, \ \ val(a_{ik}) = val(a_{jk}).$$

The two filtering rules R_i and R_j are said to be disjoint or different iff there is at least one pair of fields which are completely disjoint.

Definition 4 [Difference Relationship]

$$R_i \neq R_j \ \ iff \ \ \exists \ k \in \{1,...,m\} \ such as \ \ val(a_{ik}) \cap val(a_{jk}) = \phi.$$

This definition is considered as fundamental for the detection of anomalies in the filtering rules, and it is named the ***Fundamental Definition of Rules' Disjunction* (*FDRD*).**

The filtering rule R_i is generalized by the filtering rule R_j iff all the fields in R_i are either generalized or equal to the fields of R_j.

Definition 5 [Generalization Relationship]

$$R_i \subseteq R_j \ \ iff \ \ \forall \ k \in \{1,...,m\}, \ \ val(a_{ik}) \subset val(b_{jk}) \ ou \ \ val(a_{ik}) = val(b_{jk}).$$

The filtering rule R_i shadows the filtering rule R_j iff all the fields in R_j are either shadowed or equal to the fields of R_i.

Definition 6 [Shadowing Relationship]

$$R_i \supseteq R_j \ \ iff \ \ \forall \ k \in \{1,...,m\}, \ \ val(a_{ik}) \supset val(b_{jk}) \ ou \ \ val(a_{ik}) = val(b_{jk}).$$

It is important to notice that the order of filtering rules is very important in the identification of the types of the relationships between the filtering rules.

Definition 7 [Correlation Relationship]

$$R_i \ and \ R_j \ are \ correlated \ iff \ \exists \ H_1, H_2, H_3 \ \ such \ \ that \begin{cases} H_1 \cup H_2 \cup H_3 = H \\ H_1 \cap H_2 = H_1 \cap H_3 = H_2 \cap H_3 = \phi \end{cases}$$

$$we \ have \begin{cases} \forall a \in H_1, \ val(a_i) \subset val(a_j) \\ \forall b \in H_2, \ val(b_i) \supset val(b_j) \\ \forall c \in H_3, \ val(c_i) = val(c_j) \end{cases}$$

3. Inter-Difference Matrices

The Inter-difference Matrix (IDM) is defined, for each type of field, as a matrix which elements represent the difference of elements belonging to the correspondent vector in the matrix F.

The formal representation of the IDM matrices, called E, relatively to the vector v_k which elements are $\{a_{ik}\}_{i\in[1,n]}$, is shown in figure 2.

$$E(v_k) = (e_{ij})_{\substack{1\le i\le n \\ 1\le j\le n}} = (a_{ik} - a_{jk})_{\substack{1\le i\le n \\ 1\le j\le n}} = \begin{pmatrix} e_{11} & . & . & . & . & e_{1n} \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ e_{n1} & . & . & . & . & e_{nn} \end{pmatrix}, \quad 1 \le k \le m$$

Figure 2. *the Inter-difference matrix E*

Therefore, the IDM matrices are calculated relatively to each vector on the matrix F. Each element of the matrix represents the inter-difference of two elements on the same vector. Therefore, the set of filtering rules relationships has an algebraic structure of a semi group and by applying an isomorphism we obtain the following inter-difference coding (IDC):

- *The difference* of two fields is coded by **0**. However, the difference of two action fields is coded by **-1**.
- *The equality* of two fields -even action fields- is coded by **1**.
- *The generalization* relationship between two fields is coded by an integer **g > 0**.
- *The Shadowing* relationship between two fields is coded by an integer **s>0** (**s ≠ g).**

For the next, we set **g = 2** and **s = 3**. The table below (table 1) summarizes the IDC cases and shows the formal writing of each relationship between two fields a_{ik} and a_{jk}. The action Inter-Difference coding is treated alone.

	Formal writing	ID code	Action ID code
Shadowing	$val(a_{ik}) \supset val(a_{jk})$	**s = 3**	---
Generalization	$val(a_{ik}) \subset val(a_{jk})$	**g = 2**	---
Equality	$val(a_{ik}) = val(a_{jk})$	1	1
Difference	otherwise	0	-1

Table 1. *IDC Table*

For the next, let R_{ij} and H_{ij} be two vectors. R_{ij} is defined as the Inter-difference vector of the two rules R_i and R_j. H_{ij} is defined as the inter-difference of the two headers H_i and H_j of R_i and R_j, respectively.

The IDMs generated relatively to the vectors of the matrix F represent a model with superposed layers. Therefore, an IDM represents a layer in this model, and a vector R_{ij} is defined with the elements of all the matrices having the same position (i,j) (figure 3).

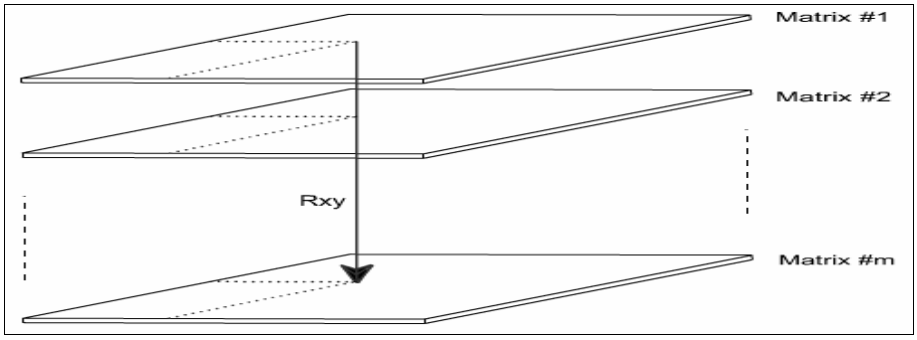


Figure 3. *IDM Layers Model*

4. Anomaly detection and identification

Before presenting the method used to detect and identify anomalies within the filtering rules, we will discuss the types of anomalies that may exist between the filtering rules.

4.1. Types of anomalies

The above study on relationships between filtering rules leads us to define the different anomalies that may exist within the filtering rules. Some of the definitions given in the following are inspired from the works of Ehab and Hazem (Hamed *et al.*, 2003*)*.

Definition 8 [Redundancy Anomaly]
The redundancy anomaly exists when there is an equality relationship between two rules with the same action.

Definition 9 [Contradiction Anomaly]
The contradiction anomaly is considered when the equality between two rules is realized and they have disjoint action fields.

Definition 10 [Generalization Anomaly]
The generalization anomaly exists between two rules if the first rule is generalized by the second rule and they have different action.

Definition 11 [Shadowing Anomaly]
The shadowing anomaly exists between two rules if the second rule is shadowed by the first rule and they have different action.

Definition 12 [Correlation Anomaly]
The correlation anomaly is considered when the correlation between two rules is realized and they have disjoint action fields.

4.2. Anomaly detection and identification Method

The IDC table is built to make easy the detection of anomalies. In fact, let x and y be the indexes of the rules R_x and R_y and let, also, H_{xy} and R_{xy} be the two inter-difference vectors respectively of the above headers and rules.

Formally H_{xy} and R_{xy} are defined as:

$$H_{xy} = (h_1, h_2, ...h_m) \ , \ where \ h_i = H_{xy}(i) = a_{xi} - a_{yi} \ , \forall i \in [1, m]$$

and

$$R_{xy} = (r_1, r_2, ..., r_{m+1}) \ , \ where \ r_i = R_{xy}(i) = h_i \ , \forall i \in [1, m] \ and \ r_{m+1} = R_{xy}(action)$$

So we can show these cases:

- **Absence of anomalies:** $\exists\, i \in [1, m] \ \ where \ \ H_{xy}(i) = 0$
- **Redundancy Anomaly:** $\begin{cases} R_{xy}(Action) = 1, \\ \forall i \in [1, m], \ H_{xy}(i) \in \{1, 2, 3\} \end{cases}$
- **Contradiction Anomaly:** $\begin{cases} R_{xy}(Action) = -1, \\ \forall i \in [1, m], \ H_{xy}(i) \in \{1\} \end{cases}$
- **Generalization Anomaly:** $\begin{cases} R_{xy}(Action) = -1, \\ \forall i \in [1, m], H_{xy}(i) \in \{1, 2\} \end{cases}$
- **Shadowing Anomaly:** $\begin{cases} R_{xy}(Action) = -1, \\ \forall i \in [1, m], \ H_{xy}(i) \in \{1, 3\} \end{cases}$
- **Correlation Anomaly:** $\begin{cases} R_{xy}(Action) = -1, \\ \forall i \in [1, m], H_{xy}(i) \in \{1, 2, 3\} \end{cases}$

It is clear that two rules which have the same action ($R_{xy}(Action) = 1$) and are not disjoint ($\forall\, i, \ H_{xy}(i) \neq 0$) present the redundancy anomaly. However, the

other anomalies exist when the values of the rule's action fields are different ($R_{xy}(Action) = -1$). Therefore, to prove the existence of anomaly, we have just to calculate the product of the elements of the vector H_{xy}, as shown in the following equation:

$$p'_{xy} = \prod_{i=1}^{m} H_{xy}(i)\ ,\ m = Fields'number\ of\ H_{xy}$$

As well, we define the product of the elements of the vector R_{xy} as the following:

$$p_{xy} = \prod_{i=1}^{m+1} R_{xy}(i)\ ,\ \ m+1 = (Fields'number\ of\ H_{xy}) + (Action)$$

Theorem

An anomaly exists between the rules R_x and R_y if and only if $p'_{xy} \neq 0$.

Proof

By the definition 4, an anomaly exists iff the rules are not disjoint then for all $i \in [1, m]$, $H_{xy}(i) \neq 0$. So $\prod_i H_{xy}(i) \neq 0$ and $p'_{xy} \neq 0$

Corollary

An anomaly exists between the rules R_x and R_y if and only if $p_{xy} \neq 0$.

Consequences

Although the product p'_{xy} permits to detect the existence of an anomaly and to identify the two rules which generate these anomaly via the indexes x and y, the product p_{xy} permits to identify also the type of anomaly. In fact, let *m* be the number of fields in the vector H_{xy}, then we have the following results:

- **Absence of anomalies**: $p_{xy} = \prod_{i=1}^{m+1} R_{xy}(i) = 0$
- **Redundancy Anomaly**: $\begin{cases} R_{xy}(Action) = 1, \\ p_{xy} = \prod_{i=1}^{m+1} R_{xy}(i) > 0 \end{cases}$,(The two rules have the same action)
- **Contradiction Anomaly**: $\begin{cases} R_{xy}(Action) = -1, \\ p_{xy} = \prod_{i=1}^{m+1} R_{xy}(i) = -(1^m) \end{cases}$, such as *m* is the number of equal fields
- **Generalization Anomaly**: $\begin{cases} R_{xy}(Action) = -1, \\ p_{xy} = \prod_{i=1}^{m+1} R_{xy}(i) = -(1^{m-j} * 2^j) \end{cases}$, such as *j* is the number of generalized fields, and *m-j* is the number of equal fields. The two rules have the same action.

- **Shadowing Anomaly**: $\begin{cases} R_{xy}(Action) = -1, \\ p_{xy} = \prod_{i=1}^{m+1} R_{xy}(i) = -(1^{m-j} * 3^{j}) \end{cases}$, such as j is the number of shadowed fields, and m-j is the number of equal fields. The two rules have the same action.

- **Correlation Anomaly**: $\begin{cases} R_{xy}(Action) = -1, \\ p_{xy} = \prod_{i=1}^{m+1} R_{xy}(i) = -(1^{m-k-l} * 2^{k} * 3^{l}) \end{cases}$, such as k is the number of generalized fields, l is the number of shadowed fields, and n-(k+l) is the number of equal fields. The two rules have the same action.

The matrix product P is composed by the different elementary products p_{xy}.

$$P = (p_{xy})_{\substack{1 \le x \le n \\ 1 \le y \le n}} = (\prod_{i=1}^{m+1} R_{xy}(i))_{\substack{1 \le x \le n \\ 1 \le y \le n}}$$

Figure 4 shows some values of p_{xy} which permit to identify each anomaly in the case of the following rule's format:

<Protocole> < Source IP > < Dest IP > <Source port> <Dest port> <Action>

The existence of the zeros in the matrix P proves the disjunction of the filtering rules, whereas the positive and negative values point out the existence of anomalies and permit to identify them.

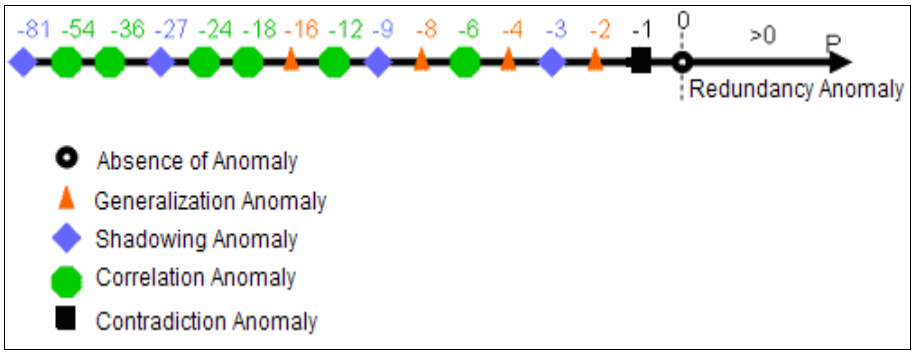


Figure 4. *Different values of the product p_{xy}*

5. Anomaly Correction

Once the anomalies have been detected and identified in the filtering rules, we discuss in this section the adequate actions which must be taken to correct the filtering rules.

In fact, the order of filtering rules which carry anomalies is a crucial propriety. Indeed, a reordering of rules, in this case, makes too difficult the understanding of the internal Firewall's working. So, we propose a model with relaxed actions. Relaxation means that we tolerate the existence of some anomalies within filtering rules. That's why we should distinguish, first, between *constructive* and *destructive* anomalies.

Constructive anomaly is defined as an anomaly which does not affect the desired filter response unless reordering is made. The two examples of constructive anomaly are the *generalization* and the *correlation*. In particular, in the case of the *correlation*, a packet whose header is the intersection of the headers of the two filtering rules in question will be treated differently according to which rule will be carried out the first, therefore the reordering of these rules will cause surely a conflict between the awaited and the real response of the filter. The other anomalies are destructive because they can produce an unexpected response of the filter even without reordering. Explicitly, in the case of the shadowing or the contradiction, the second rule in order will never be executed. The redundancy is also a destructive anomaly because it increases the number of rules which decreases the performance of the filter.

Consequently, relaxation consists in tolerating the existence of the constructive anomalies and to treat carefully their order, then to apply relaxed actions to correct the destructive anomalies. These actions are:

Recommended rule's removal: this action is applied in the case of the redundancy anomaly. In fact, it is recommended to delete the specific rule which is included in the other rule.

Commanded rule's removal: this action is employed when *contradiction anomaly* is detected. Administrator commands it and chooses which rule will remove.

Recommended rule's permutation: this action is executed in the case of shadowing. The permutation of the two rules induces the anomaly generalization which is tolerable by this relaxed model.

Commanded rule's permutation: *this* action is applied in the case of the correlation. Though this anomaly is allowed in the relaxed correction, we must ask the administrator which rule has to be placed the first.

In conclusion, all these actions aim to have a "stable" filter in order to guarantee the desired and correct output of the firewall. The filter obtained reflects accurately the security policy required.

6. Dealing with the reordering issue

Our model permits to detect, identify and to correct the anomalies which can exist within filtering rules. We propose in this section to study the issues of the reordering which can appear after the correction namely.

The problem of the reordering exists since certain Firewalls sort rules during the storage of the filtering list. And as the saved rules are inter-related, any modification of the initial order of the rules will affect the desired response of the filter.

In fact, in our relaxed case, the reordering is a main problem. Constructive anomalies are likely to be destructive. Thus in order to solve this problem, we will apply our matrix model to restore again the initial order of the filter.

The starting point of this approach is the property of the relaxed model which consists in tolerating the existence of the generalization and correlation anomalies. The other anomalies are eliminated by correction. Therefore, in the matrix *P*, we will only find the characteristic values of these anomalies. If we represent the reordering by a transformation *T*, then the resulting matrix $P' = T(P)$ obtained after the transformation includes new values corresponding to the *shadowing anomaly* and *correlation anomaly*, possibly different from those in the matrix *P*. Consequently, we can apply the *commanded permutation* and *the recommended* actions of the relaxed correction to restore the initial organization.

7. Dealing with the insertion issue

The insertion of a new rule represents the second issue to be solved in this paper. Indeed, adding a new rule can affect the stability of the filter and reveals anomalies with the other rules. Thus our objective is to seek the best order of the new rule to minimize the number of anomaly and to stabilize the filter.

The method which will be used is the derivative of our matrix method. Indeed, we only restrict the work to calculate the inter-difference vectors characterizing the relationships of each field of the new rule with the corresponding fields of each rule on the filter. Formally, if *k* indicates the field's index and *N* the number of rule in the filter, then the inter-difference vector *V* is written:

$$V(k) = (e_i(k))_{1\le i\le n} = (R(k) - R_i(k))_{1\le i\le n} = (e_1(k),...,e_n(k))^T, \quad \forall k = 1..m$$

Such as R is the rule to be inserted and $\{R_i\}$ the set of the filtering rules.

Once *V(k)* are performed, we calculate the product vector *P″* defined by the following formula:

$$P'' = (p_i'')_{1\le i\le n} = (\prod_{k=1}^{m} e_i(k))_{1\le i\le n} = (p_1'',..., p_n'')^T \quad , \quad m: number\ of\ field$$

The vector *P″* permits to detect and identify the anomalies generated by the insertion of the new rule R. therefore, we apply the algorithm of the relaxed

correction. Several permutations will be made with an aim of having only the *generalization* and *the correlation* anomalies. A table of traces of the permutations is necessary to be able to apply them to the real filter. The process of the correction stops when the vector *P''* presents only the values characteristic of the *generalization* and *correlation* anomalies.

9. Conclusion

In this paper, we have proposed a new matrix approach for detecting anomalies in the Firewall filtering rules. The filtering list is represented by a matrix. Then, we have generated a number of matrices defining all the relationships between rules. Our approach works with any type of rules. Once the anomaly is detected, we propose a method based on the matrices generated which permits to identify its type and the corresponding rules. Then, we have proposed a set of actions in order to correct the anomalies detected in the filtering rules. Two issues which can arise after the correction are treated. First we have discussed the issue of the reordering and we have shown that based on our matrix approach we can restore the initial order of the filtering rules. Secondly, we have treated the issue of the insertion of a new rule. And we have demonstrated also that we can find the optimum order of the inserted rule by the application of our matrix approach.

References

B. Hari, S. Suri and G. Parulkar. “Detecting and Resolving Packet Filter Conflicts.” Proceedings of IEEE INFOCOM’00, Mars 2000.

Hazem Hamed and Ehab Al-Shaer : “Management and Translation of Filtering Security Policies”, IEEE ICC'03, May 2003

Ehab S. Al-Shaer and Hazem H. Hamed : “Firewall Policy Advisor for Anomaly Discovery and Rule Editing”. IEEE/IFIP Integrated Management (IM'2003), March 2003.

Pasi Eronen and Jukka Zitting: “An expert system for analyzing Firewall rules”. Helsinki University of Technology

Vadim Zaliva , Vadim Kurland : “Firewall Builder Architecture Overview” .

T.K. Lee, S. Yusuf, W. Luk, M. Sloman, E. Lupu and N. Dulay : “Development Framework for Firewall Processors”. Department of Computing, Imperial College, England.

Joshua D. Guttman. Filtering postures: Local enforcement for global policies. In Proceedings of the 1997 IEEE Symposium on Security and Privacy, Oakland, California, May 1997.

Adiseshu Hari, Subhash Suri, and Guru Parulkar. Detecting and resolving packet filter conflicts. In Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000.

Scott Hazelhurst. Algorithms for analysing firewall and router access lists. Technical Report TR-Wits-CS-1999-5, Department of Computer Science, University of the Witwatersrand, South Africa, July 1999.

Scott Hazelhurst, Adi Attar, and Raymond Sinnappan. Algorithms for improving the dependability of firewall and filter rule lists. In Proceedings of the International

Conference on Dependable Systems and Networks (DSN 2000)New York, June 2000. IEEE Computer Society Press.

Scott Hazelhurst, Anton Fatti, and Andrew Henwood. Binary decision diagram representations of firewall and router access lists. Technical Report TR-Wits-CS-1998-3, Department of Computer Science, University of the Witwatersrand, South Africa, October 1998.

Mohsen Razvani and Rasool Jalili. Specification and Verification of Security Policies in Firewalls. Department of Computer Engineering, Sharif University of Thechnology, Tehran, Iran.

D. Brent Chapman et Elizabeth D. Zwicky : « Firewalls », traduction de Jean Zundel, Janvier 2002.

S.M.Belloven et W.R.Cheswick : « Firewalls et sécurité Internet », traduction de catherine Reclus, Octobre 1995.

John Wack, Ken Cutler, Jamie Pole: "Guidelines on Firewalls and Firewall Policy: Recommendations of the National Institute of Standards and Technology"

Christoph L. Schuba. On the Modeling, Design, and Implementation of Firewall Technology. Doctoral dissertation, Purdue University, December 1997.

Appendix 1: Example of detecting anomalies based on the proposed approach

Let F be the following set of filtering rules:

order	Protocol	Source Address	Dest Address	Source Port	Dest Port	Action
1	TCP	140.192.37.20	*.*.*.*	***	80	Deny
2	TCP	140.192.37.*	*.*.*.*	***	80	Accept
3	TCP	*.*.*.*	161.120.33.40	***	80	Accept
4	TCP	140.192.37.*	161.120.33.40	***	80	Deny
5	TCP	140.192.37.30	*.*.*.*	***	21	Deny
6	TCP	140.192.37.*	*.*.*.*	***	21	Accept
7	TCP	140.192.37.*	161.120.33.40	***	21	Accept
8	UDP	140.192.37.*	161.120.33.40	***	53	Accept
9	UDP	*.*.*.*	161.120.33.40	***	53	Accept
10	*	*.*.*.*	*.*.*.*	***	***	Deny

Table 2. *Filter F*

Therefore, based on our approach, we start the process of detecting anomalies by performing the six IDM matrices relatively to each field. The first matrix is given by the following figure, and it shows the different relationships existing between all elements of the protocol vector. Indeed, the blocs of 1 indicate the equality of some elements of the vector protocol, and the number of blocs reflects the number of different values of the protocol field. In fact, we find two blocs of 1 which show that we have two values of the protocol field namely TCP and UDP.

1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	0	1	1

Protocol Matrix

Then, the following figures show the IDM matrices relatively to the source port and the destination port. Although, the first matrix has all its elements equal to 1, the second matrix, describing the relationships between the elements of the destination port vector, presents 3 blocs of 1 and the other values are equal to zero.

1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1

Source port Matrix

1	1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	0	0
0	0	0	0	1	1	1	0	0
0	0	0	0	1	1	1	0	0
0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	0	1	1

Destination port Matrix

The IDMs of the source address and the destination address are given by the following figures. We can shows in these matrices all the coding values {0,1,2,3}.

1	2	2	2	0	2	2	2	2
3	1	2	1	3	1	1	1	2
3	3	1	3	3	3	3	3	1
3	1	2	1	3	1	1	1	2
0	2	2	2	1	2	2	2	2
3	1	2	1	3	1	1	1	2
3	1	2	1	3	1	1	1	2
3	1	2	1	3	1	1	1	2
3	3	1	3	3	3	3	3	1

Source address Matrix

1	1	3	3	1	1	3	3	3
1	1	3	3	1	1	3	3	3
2	2	1	1	2	2	1	1	1
2	2	1	1	2	2	1	1	1
1	1	3	3	1	1	3	3	3
1	1	3	3	1	1	3	3	3
2	2	1	1	2	2	1	1	1
2	2	1	1	2	2	1	1	1
2	2	1	1	2	2	1	1	1

Destination address Matrix

Finally, the following figure describes the IDM of the action field. We can show the presence of the two values 1 and -1 used to code the relationships in the case of the action field.

1	-1	-1	1	1	-1	-1	-1	-1
-1	1	1	-1	-1	1	1	1	1
-1	1	1	-1	-1	1	1	1	1
1	-1	-1	1	1	-1	-1	-1	-1
1	-1	-1	1	1	-1	-1	-1	-1
-1	1	1	-1	-1	1	1	1	1
-1	1	1	-1	-1	1	1	1	1
-1	1	1	-1	-1	1	1	1	1
-1	1	1	-1	-1	1	1	1	1

Action Matrix

Once all these matrices are performed, we can now generate the matrix product P given by the following figure.

1	-2	-6	6	0	0	0	0	0
-3	1	6	-3	0	0	0	0	0
-6	6	1	-3	0	0	0	0	0
6	-2	-2	1	0	0	0	0	0
0	0	0	0	1	-2	-6	0	0
0	0	0	0	-3	1	3	0	0
0	0	0	0	-6	2	1	0	0
0	0	0	0	0	0	0	1	2
0	0	0	0	0	0	0	3	1

Matrix P (Anomalies detection)

According to the matrix P, we can see that the existing values, except zero, reflect an anomaly. Explicitly,

0: absence of anomalies between the rules in question.

1: this value is not considered as an indication of the redundancy anomaly because it reflects the relationships of between one rule and itself.

2, 3, and 6: these values prove the existence of the *redundancy anomaly* between the rules **1** and **4**, **2** and **3**, **6** and **7** and finally **8** and **9**.

-2: this value proves the existence of the *generalization anomaly* between the rules **1** and **2**, and **5** and **6**.

-3: this value proves the existence of the *shadowing anomaly* between the rules **2** and **4**, and **3** and **4**.

-6: this value proves the existence of the *correlation anomaly* between the rules **1** and **3**, and **5** and **7**.

Correction:

By the application of our modal for the correction of the anomalies, the resulting filter is shown by the following figure:

order	Protocol	Source Address	Dest Address	Source Port	Dest Port	Action
1	TCP	140.192.37.20	*.*.*.*	***	80	Deny
2	TCP	140.192.37.*	161.120.33.40	***	80	Deny
3	TCP	140.192.37.*	*.*.*.*	***	80	Accept
4	TCP	*.*.*.*	161.120.33.40	***	80	Accept
5	TCP	140.192.37.30	*.*.*.*	***	21	Deny
6	TCP	140.192.37.*	*.*.*.*	***	21	Accept
7	UDP	*.*.*.*	161.120.33.40	***	53	Accept
8	*	*.*.*.*	*.*.*.*	***	***	Deny

Table 3. *Corrected Filter*

Appendix 2: Algorithm of the detection and the identification of anomalies

```
Void AnomalyDetection (matrix F, int vector_nbre, int rules_nbre)
{
        int m = vector_nbre   ;
        int  n = rules_nbre;
//Performing the Inter-Differece Matrices
for (int k=  0 ; k < m ; k++)
                for (int i=  1 ; i < n+1 ; i++)
                for (int j=  i+1 ; j < n+1 ; j++)
                                E[k][i][j] =  F[i][k] – F[j][k];
// E[k] is the IDM related to each vector
      //Generating the matrix P to detect anomalies
for (int i=  1 ; i < n ; i++)
        for (int j=  i+1 ; j < n+1 ; j++)
        {
                P[i][j]=1;
                for (int k=  0 ; k < m ; k++)
                        P[i][j]*= E[k][i][j];
                }
// the following code is to identify the anomalies
for (int i=0 ; i<n ; i ++)
              for (int j=i+1; j<n; j ++)
                {
                   If (P[i][j] = = 0) then Absence_Anomaly;
                   Else
                 if (P[i][j] >0) then Redundance_Anomaly;
                        Else
                        if (P[i][j] mod 6 = = 0) then Correlation_Anomaly;
                                Else
                                if (P[i][j] mod 2 = = 0) then
                                Generalization_Anomaly;
                                        Else
                                         if (P[i][j] mod 3 = = 0) then
                                        Shadowing_Anomaly;
                }
}
```

Approche langage pour la spécification des politiques de sécurité

Hamdi Hédi

Labri 351, cours de la Libération
33405 Talence cedex
hamdi@labri.fr

Adel Bouhoula et Mohamed Mosbah

Cité Technologique des Communications
Rte de Raoued Km 3,5 2083, Ariana Tunisie
bouhoula@planet.tn

Labri 351, cours de la Libération
33405 Talence cedex
mosbah@labri.fr

RÉSUMÉ *Une politique de sécurité prescrit les actions acceptables, relatives à la sécurité dans un système d'information. Il y a, donc un besoin pressant pour les organisations de définir et de spécifier leurs politiques de sécurité, même d'une manière informelle. Cependant, la spécification formelle ou automatique d'une politique de sécurité présente plusieurs avantages. En effet, elle permet la vérification de la cohérence et de la complétude des politiques, leur mise en œuvre automatique ainsi que d'autres opérations formelles telles que la composition des politiques. Nous introduisons dans ce papier, un nouveau langage nommé PPL (Policy Programming Language) pour la spécification des politiques de sécurité. Ce langage permet la spécification déclarative de haut niveau des politiques de sécurité, et permet d'effectuer plusieurs vérifications spécifiques au domaine de la sécurité.*

ABSTRACT. *A security policy, states the acceptable actions, related to security in an information system. Therefore, there is a pressing need for organizations to specify their security policies, even in an informal way. However, the formal or automatic specification of a security policy has several advantages. Indeed, it enables the verification of the coherence and the completeness of the policies, their automatic implementation as well as other formal operations such as the composition of the policies. We present in this paper a new programming language, named PPL, for specifying security policy. PPL offers high-level and declarative constructs; it improves robustness by enabling a variety of verifications to be performed.*

MOTS-CLÉS : politique, langage, verification, analyse, anomalie, conflit
KEYWORDS: policy, language, verification, anomaly, conflict

1. Introduction

Avec le développement des réseaux, l'interopérabilité et la portabilité toujours plus importantes d'applications distribuées différentes et l'utilisation massive de l'informatique pour la manipulation des informations sensibles, la sécurité des systèmes informatiques est de plus en plus au coeur des préoccupations.

En effet, dès l'apparition des systèmes informatiques la sécurité des données manipulées et stockées est apparue comme un problème important. Cependant, la prise en compte de la sécurité dans un environnement distribué est un vrai défi, à cause de l'incompatibilité des technologies utilisées par les différentes plates-formes distribuées (e.g. EJB, .NET, etc). Plusieurs applications distribuées fonctionnent actuellement sans sécurité, ou alors utilisent des méthodes ad hoc, pour éviter l'incompatibilité des techniques de sécurité entre plate-formes hétérogènes.

Depuis une décennie environ, la spécification de la sécurité par les politiques est devenue un thème de recherche actif.

Cet intérêt est dû au fait qu'elle permet :

- de fournir des solutions de sécurité auto-adaptables, qui peuvent changer dynamiquement le comportement du système contrôlé.
- de proposer des solutions qui supportent l'adaptabilité dynamique du comportement en changeant la politique sans arrêter le système
- de faciliter la mise à jours dynamique des règles de politique interprétées par les entités distribuées pour modifier leur comportement.

Dans cet article, nous proposons une nouvelle approche pour la représentation des politiques de sécurité, que nous présentons sous la forme d'un langage dédié à la spécification des politiques de sécurité, ce langage est :

Spécifique au domaine : La conception et le développement de PPL ont été basés sur une analyse approfondie du domaine de la sécurité informatique. Cette analyse comprenait l'étude de plusieurs outils de spécification existants et d'exemples de politiques de sécurité typiques

Haut niveau : Notre langage offre des constructions et des types de données de haut niveau qui permettent aux programmeurs d'exprimer et de spécifier des politiques de sécurité cohérentes.

Déclaratif : avec PPL on spécifie uniquement les règles et les actions d'une politique; le compilateur traduit la spécification en une implémentation efficace et génère automatiquement les moniteurs nécessaires (par exemple un seul firewall ou plusieurs dont chacun contrôle une partie du réseau). Les informations nécessaires à la vérification et l'analyse de la politique sont capturées dans le programme.

Robuste: PPL est plus sûr qu'un langage généraliste car sa syntaxe et sa sémantique permettent des vérifications spécifiques au domaine. En particulier, le compilateur PPL vérifie la cohérence de la composition des règles de sécurité.

Les contributions de notre travail sont les suivantes :

- Nous avons identifié les aspects communs et les concepts clés utilisés dans les politiques de sécurité, on nous basant sur l'étude de plusieurs exemples.
- Nous présentons la définition de PPL, un langage déclaratif de haut niveau dédié à la spécification des politiques de sécurité. Le langage est fortement typé et permet différentes vérifications de cohérence.
- Nous montrons que PPL est expressif, il a été utilisé dans la spécification de plusieurs types de politiques de sécurité.
- Nous démontrons que PPL est concis. Notre spécification PPL est beaucoup plus petite que son équivalent écrit dans un autre langage tel que ponder (Damianou, 2002).
- Nous avons implémenté un compilateur pour PPL. Les configurations générées sont plus efficaces que celles effectuées manuellement.

Ce papier est organisé de la manière suivante. La section 2 présente les approches de mise en oeuvre de la sécurité informatique. Dans la section 3 nous introduisons le langage PPL en nous focalisant sur ses principales abstractions. La section 4 donne une description du processus de compilation. La section 5 liste les principales vérifications effectuées dans un programme PPL. Nous présentons les travaux connexes dans la section 6 puis nous concluons et évoquons nos futurs travaux dans la section 7.

2. Approches de mise en œuvre de la sécurité informatique

La sécurité informatique consiste aujourd'hui en grande partie en des méthodes défensives employées pour détecter et contrecarrer les possibles intrus et parer aux différents types de menaces possibles.

Il y a principalement quatre grandes approches à la réalisation d'un environnement sécuritaire : l'utilisation de procédures spéciales à suivre, l'ajout de fonctions ou mécanismes additionnels dans le système, l'emploi de techniques de vérification dans le but d'augmenter la confiance placée dans sa sécurité, et l'utilisation de systèmes de détection d'intrusion. Ces approches sont générales dans le sens où elles ne sont pas dédiées à un type de système en particulier. Ci-après nous présentons brièvement ces quatre approches

2.1. *Les approches procédurales*

En partant des besoins de sécurité exprimés pour un système, il est possible de les satisfaire en exigeant que certaines procédures soient suivies par l'utilisateur du

système. Les approches procédurales prescrivent le comportement approprié à suivre par l'utilisateur lorsque celui-ci utilise le système.

2.2. *Les fonctions et mécanismes supplémentaires*

L'ajout dans le système de mécanismes ou fonctions supplémentaires qui forcent le respect des exigences de sécurité constitue une autre approche à la sécurité informatique. Ces mécanismes sont les mécanismes d'authentification, de contrôle d'accès et de flux d'information.

2.3. *Les techniques de vérification*

Cette approche soumet un système aux techniques de vérification rigoureuses et permet ainsi d'augmenter la confiance en un comportement sécuritaire de celui-ci. Parmi ces techniques l'analyse de pénétration, la spécification et la vérification formelles et l'analyse des canaux cachés. Ces méthodes augmentent seulement la confiance placée en la sécurité du système sans la garantir.

2.4. *La détection d'intrusion*

Les systèmes de détection d'intrusion définissent des moyens de protection évolutifs et flexibles. Ils sont utilisés pour détecter des tentatives d'intrusion, internes ou externes, visant à porter atteinte à la sécurité des systèmes.

3. Le langage PPL

La conception du langage PPL a été guidée par une analyse approfondie du domaine de la sécurité informatique en général, et essentiellement celui de la sécurité à base de politiques (Thibault, 1998). Nous avons examiné divers types d'outils de spécification de politiques de sécurité connues, de politiques de sécurité typiques (IPsec (Wolfien, 2002)), ainsi que d'autres plus dédiés (politiques de sécurité pour les services Web (Finin *et al*, 2002), politiques de sécurité pour le Web sémantique (Finin *et al*, 2000), politiques de sécurité pour le système d'information de la santé (Andreson, 1996)), plates-formes et outils de spécification de politiques de sécurité (Ponder (Damianou,2002), PDL(Bhatia *et al*, 1999)), et diverses documentations et articles. En se basant sur cette analyse de domaine, nous avons identifié les caractéristiques suivantes pour un langage dédié à ce domaine. Le langage doit être composé de cinq blocs de base : entité, scope, règle, action et politique pour décrire les opérations appropriées pour assurer la sécurité d'un système donné. Il doit inclure des déclarations de blocs spécifiques pour permettre des vérifications et des analyses dédiées. Il doit être modulaire pour permettre de décomposer une spécification de politique de sécurité dans des composants maniables, et les composer dans des politiques plus complexes jusqu'à former une politique globale. Il doit contenir des interfaces qui permettent la réutilisation systématique de bibliothèques d'actions de sécurité.

3.1. *Présentation du langage PPL*

Un programme PPL définit une liste de blocs. *Les déclarations de blocs* décrit quel sujet (utilisateur ou processus) peut accéder à un objet et sous quelles conditions. Un bloc peut être ou bien *une politique, une règle, une action* ou *une entité, un scope. le scope* représente la liste des entités impliquées dans la politique. *Une politique* correspond à une séquence de règles permettant de déterminer des paramètres de configuration spécifiques à un niveau de protection, elle peut être simple ou composée. *Une règle* consiste en un ensemble de contraintes sur un ensemble d'actions, elle peut être ou bien « simple déclenchement » quand uniquement une seule action est déclenchée pour un objet donné, ou « multi déclenchement » quand différentes actions sont déclenchées pour le même objet. *Une action* peut être atomique ou composée. Dans ce qui suit nous présentons en détail chacun de ses blocs de base et nous montrons comment ils sont utilisés pour spécifier une politique de sécurité en PPL.

3.2. Politique

Une politique en PPL est un ensemble de règles et de scopes qui gouvernent les événements particuliers d'un domaine. Ces règles définissent le comportement idéal d'un système(Sloman, 1994). PPL supporte une série extensible de types de politiques de sécurité.

Syntaxiquement on définit une politique de sécurité comme suit :

<type>policy***<name>*** ***{***

scope *::= <entity list>;*

body*::= <policy_rule list> ; }*

La construction **scope** définit sur quel domaine la politique est appliquée. Il liste toutes les entités du système qui sont concernées par la politique en question. Le **body** implémente la politique via un ensemble de règles de sécurité.

Le *type* permet de distinguer le type de la politique. On dénombre :

- *Politique d'autorisation :* ce type de politique est lié au contrôle d'accès (Sandhu *et al*,1994) (Abrams , 1993), il définit une séquence d'actions qu'un sujet est autorisé ou interdit d'exécuter sur un ensemble d'objets cibles. Cette politique permet de protéger les objets cibles et elle est généralement interprétée par le système de contrôle d'accès.
- *Politique d'obligation :* déclenchée généralement par des événements, elle spécifie une séquence d'actions qu'un sujet doit exécuter sur un ensemble d'objets cible, en réponse à un événement particulier. Elle définit les devoirs des sujets dans le scope de la politique
- *Politique de délégation :* cette politique spécifie les actions et les privilèges qu'un sujet peut déléguer à un autre. Une politique de délégation définit ainsi une autorisation à déléguer.

- *Politique composée :* utilisée pour grouper un ensemble de spécifications de politiques de sécurité pour simplifier la tâche de spécification de politique pour un système distribué.

3.3. *Règle*

Consiste en un ensemble d'actions que les sujets peuvent exécuter sur un ensemble d'objets cibles quand un ensemble de contraintes est vérifié. Dans une règle « simple déclenchement », une seule action est déclenchée sur un même objet quand une condition est satisfaite, alors que dans une règle « multiple déclenchement », différentes actions sont déclenchées sur un même objet quand une condition est satisfaite.

La syntaxe d'une règle de sécurité est la suivante :

rule<*name*><*parameter list*> *{*
subject: < *subject list*>;
object: < *object list*>;
if <*condition*>
then < *action list*>; }

3.3. *Action*

Une action correspond à une opération élémentaire ou composée du système qui modifie son état (exemples : les opérations de création et destruction d'un objet ou d'un sujet ou encore la modification ou la délégation d'un droit d'accès). Les actions définissent quelles modifications un sujet peut accomplir sur un ensemble d'objets appartenant à un domaine ciblé.

La syntaxe d'une action se présente comme suit :

action<*name*>< *parameter list* >{ <*operation list*>}

3.4. *Entité*

Les entités en PPL sont des objets typés avec des interfaces explicites permettant d'extraire leurs propriétés. Une entité peut être un objet, un sujet ou une collection d'objets et de sujets.

3.5. *Scope*

C'est une collection d'entités. Le scope est essentiel pour toute politique de sécurité pour assurer la compacité, la généralisation et l'adaptabilité. Le scope permet d'éviter qu'une règle soit répétée pour chaque entité sur laquelle est elle appliquée, et permet de simplifier la gestion d'un grand nombre d'entités. PPL offre deux types de scopes : classe et domaine. Une classe est un ensemble d'entités classées par rapport à leurs propriétés, par exemple, tous les paquets TCP. Un domaine est un ensemble défini par extension (ajout et suppression explicite d'éléments). La syntaxe d'un scope se présente comme suit :

scope<*name*>{<*entities list*>}

4. Compilation

Nous avons développé un compilateur pour PPL, qui génère automatiquement des moniteurs de sécurité pour un réseau donné. Le processus de compilation dans PPL comprend les étapes suivantes. D'abord, les dépendances entre les blocs sont collectées et représentées sous forme de liste. Puis cette liste est annotée par les informations sur actions à entreprendre. Ensuite la liste résultante est utilisée pour extraire les règles cohérentes. En fin les configurations ou les moniteurs d'implémentation de la politique sont générés.

5. Vérifications

5.1. Détection de conflits et d'anomalies :

PPL étant un langage dédié à la spécification des politiques de sécurité, son compilateur permet d'effectuer plusieurs vérifications. Ces vérifications concernent essentiellement la détection de conflits entre les règles de sécurité dans une politique (les conflits intra-politique), et les conflits entre les politiques (les conflits inter-politiques) ainsi que la détection d'anomalies. Les conflits sont surtout des conflits de modalités et des conflits sémantiques:

- **Conflits de modalités** : L'analyse d'une spécification de politique de sécurité permet d'énumérer pour l'ensemble des règles de la politique les tuples (sujet, objet, action) sur lesquels sont appliquées ces règles. S'il y a deux règles ou plus qui sont appliquées a un même tuple (sujet, objet, action), alors il y a un conflit potentiel et la politique doit être vérifiée pour voir s'il y a un conflit réel (par exemple, une règle d'autorisation et une règle d'interdiction appliquées aux même tuple (sujet, objet, action)). Sommairement, les conflits de modalité se classifient comme suit : **le conflit d'autorisation** qui se produit quand deux règles d'autorisation positive et négative sont définies pour les mêmes sujets, objets et actions, **le conflit d'obligation** quand une règle oblige un sujet à exécuter une action et pendant qu'une autre règle l'oblige à faire l'inverse et le **conflit d'obligation non autorisée** qui se produit quand une règle oblige un sujet à faire une action qu'il n'a pas l'autorisation de faire.
- **Conflits sémantiques :** Les conflits sémantiques sont des conflits provoqués par des contraintes spécifiques aux applications auxquelles est appliquée la politique de sécurité. Par exemple, un système qui implémente le principe de séparation des pouvoirs et des devoirs, déclare l'existence d'un conflit quand une personne qui a passé un examen va elle-même le corriger ou lorsque une personne valide un chèque qu'elle va encaisser.

Le processus de vérification permet aussi de détecter des anomalies. Nous présentons ci-après ces anomalies suivies par un exemple de firewalls distribués qui montre comment elles seront détectées avec PPL.

- **Anomalie de généralisation :** Une règle est une généralisation d'une autre, si elles ont des actions différentes et la première règle est appliquée aussi sur tous les objets de la deuxième.
- **Anomalie de recouvrement :** Une règle est recouverte quand une règle précédente est appliquée sur tous ses objets, sachant que les deux règles ont des actions différentes.
- **Anomalie de corrélation :** Deux règles avec des actions différentes, sont corrélées si chacune d'entre elles est aussi appliquée sur un sous-ensemble des objets sur laquelle l'autre est appliquée.
- **Anomalie de redondance :** Une règle est redondante si elle exécute la même action sur les mêmes objets qu'une autre règle.

5.2. *Exemple : Firewalls distribués*

Un réseau d'une université contient trois sous réseaux nommés respectivement LAN1, LAN2 et LAN3 (figure 1). LAN1 contient les serveurs qui offrent des services aux utilisateurs externes, ces serveurs sont un serveur DNS, un serveur mail, un serveur Web et un serveur FTP avec aussi un certain nombre de machines. LAN2 est le sous réseau des enseignants, il est composé d'un serveur FTP interne ainsi que des machines. LAN3 est le sous réseau des étudiants.

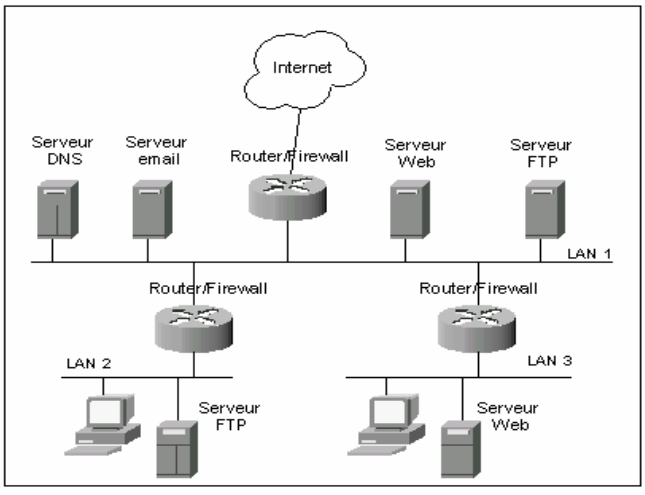


Figure1 : architecture du réseau

Sachant que :

- La plage d'adresses pour LAN1 est 193.95.30.X.
- La plage d'adresses pour LAN2 est 172.16.50.X
- La plage d'adresses pour LAN3 est 172.16.16.X.
- L'adresse du serveur DNS du FAI est : 193.95.60.10.
- L'adresse du serveur SMTP du FAI est : 193.95.60.20.

5.2.1. Politique de sécurité

La politique de sécurité adoptée pour ce réseau, contient les règles suivantes :

1. *le serveur DNS est autorisé uniquement à accéder à un serveur DNS externe d'un FAI (fournisseur d'Accès Internet)*
2. *le serveur SMTP est autorisé uniquement à accéder à un serveur mail externe d'un FAI*
3. *le serveur Web et le serveur FTP sont accessibles par tout utilisateur externe*
4. *les utilisateurs du LAN2 sont autorisés à accéder au serveur Web du LAN3*
5. *les utilisateurs du LAN3 ne sont pas autorisés à accéder au serveur FTP du LAN2*
6. *les utilisateurs du LAN3 ne sont pas autorisés à faire des « ping » sur des machines dans LAN2*
7. *les utilisateurs du LAN2 sont autorisés à faire des « ping » sur des machines dans LAN1 et LAN3*
8. *les utilisateurs externes ne sont pas autorisés à faire des « ping » sur des machines dans LAN1, LAN2 et LAN3*
9. *Les utilisateurs externes et internes ne sont pas autorisés à faire des « ping » sur des machines dans LAN2, à l'exception des utilisateurs de LAN1*
10. *tout contrôle de trafic ICMP, provenant d'une machine externe est interdit*
11. *Les machines du LAN3 ne sont pas autorisées à envoyer des erreurs de trafic ICMP.*
12. *toute machine du LAN1, LAN2 et LAN3 est autorisée à envoyer des erreurs de trafic ICMP de type 3*
13. *toute machine du LAN3 est autorisée à envoyer des erreurs de trafic ICMP de type 13 aux machines externes*
14. *toutes les machine du LAN1, LAN2 et LAN3 ne sont pas autorisées à envoyer des erreurs de trafic ICMP aux machines externes*

La spécification de cette politique en PPL est donnée en annexe

L'implémentation de cette politique en PPL génère les tables de filtrage suivantes avec la liste des anomalies détectées. Nous présentons par la suite les tables de filtrage générées pour chaque Firewall ainsi que la liste des anomalies détectées.

5.2.2. *Règles de filtrage pour le firewall du LAN1*

Trafic TCP :

Règle	**Direction**	**IP source**	**IP destination**	**Port source**	**Port destination**	**action**
1	Outgoing	193.95.30.1	193.95.60.10	Any	53 (DNS)	Allow
2	Outgoing	193.95.30.2	193.95.60.20	Any	25 (SMTP)	Allow
3	Incoming	Any	193.95.30.3 and 193.95.30.4	Any	21(ftp) or 80(http)	Allow

Trafic ICMP :

Règle	**Direction**	**IP source**	**IP destination**	**type**	**code**	**action**
8	incoming	Any of external hosts	any	8	0	Deny
10	incoming	Any of external hosts	any	0,8, 9,10,13,14,15,16,17,18	Any number	Deny
12	outgoing	Any of external hosts	any	3	0	Accept
14	Outgoing	Any of external hosts	any	0,8, 9,10,13,14,15,16,17,18	0	Deny

5.2.3. *Règles de filtrage pour le firewall du LAN2*

Trafic TCP :

Règle	**Direction**	**IP source**	**IP destination**	**Port source**	**Port destination**	**action**
4	Outgoing	any	172.16.16.X	Any	80(http))	Allow

Trafic ICMP :

Règle	Direction	IP source	IP destination	type	code	action
7	Outgoing	Any	Any IP of LAN1 and LAN3	8	0	Allow
9.1	incoming	Any of external hosts	Any IP of LAN2	8	0	Deny
9.2	incoming	Any of LAN1	Any of LAN2	8	0	Allow
9.3		Any of LAN3	Any of LAN2	8	0	Deny

5.2.4. *Règles de filtrage pour le firewall du LAN3*

Trafic TCP :

Règle	Direction	IP source	IP destination	Port source	Port destination	action
5	Outgoing	any	172.16.50.1	Any	21	Deny

Trafic ICMP :

Règle	Direction	IP source	IP destination	type	code	action
6	Outgoing	Any	Any IP of LAN2	8	0	deny
11	outgoing	Any	Any	3, 4, 5, 9, 11,12	any	deny
13	outgoing	Any	Any	13	Any	Accept

5.2.5. *Liste des anomalies détectées*

La phase de vérification et d'analyse dégage pour cette politique les anomalies suivantes :

- la règle n°8 est redondante avec la règle 10 dans la configuration du firewall du LAN1. Elle doit être supprimée.
- la règle n°9.1 de la configuration du firewall du LAN2 est généralisée par la règle n°8 du firewall du LAN1, et par conséquent elle peut être supprimée.
- La règle n°9.1 de la configuration du firewall du LAN2 est généralisée par la règle n°10 du firewall du LAN1, et par conséquent elle peut être supprimée.

- La règle n°11 de la configuration du firewall du LAN3 est recouverte par la règle n°12 du firewall du LAN1, et par conséquent on doit réviser la politique de sécurité.
- La règle n°14 de la configuration du firewall du LAN1 est recouverte par la règle n°13 du firewall du LAN3, et par conséquent on doit réviser la politique de sécurité.

6. Travaux connexes

Le langage de spécification *Ponder* mis au point par Damianou et al.(Damianou *et al*, 2001), est certainement le travail le plus en relation avec le nôtre. Même si nous partageons les mêmes objectifs, nos deux approches varient considérablement. *Ponder* est essentiellement un langage généraliste, qui est formé d'un sous-ensemble de Java. Les performances d'une spécification compilée de *Ponder* sont comparables à ce que produisent les compilateurs existants de Java, qui restent beaucoup plus lents même en comparaison à d'autres langages généralistes. L'utilisation du code java pour la spécification de politiques de sécurité ne permet pas d'effectuer les vérifications spécifiques au domaine. Une autre différence entre PPL et *Ponder* est relative à la pertinence pour spécifier la sécurité, c'est l'incapacité de *Ponder* d'exprimer certains aspects liés à la communication tels que la confidentialité et l'intégrité. Ces aspects sont permis en PPL par la possibilité d'exprimer un ensemble de paramètres nécessaires pour la sécurité des communications, par exemple la source et la destination d'un message. Ponder est conçu pour être un langage de politique, cependant au niveau réseaux, lorsque des interactions avec des models standards de politique est nécessaires, une approche à base de règle est plus intéressante, car la translation est beaucoup plus facile. C'est pour cela nous avons conçu PPL de manière à mixer conjointement les deux approches, en effet PPL fournit une flexibilité, une abstraction et une complétude dans la spécification de politiques et en même temps permet de spécifier les règles. Un autre atout de PPL par rapport à *Ponder* concerne les mécanismes de translation que fournit PPL et qui permettent de passer automatiquement d'un niveau d'abstraction à un autre.

Il existe plusieurs autres langages de spécification de politiques de sécurité, plus ou moins proches de notre langage. Le langage ASL (Authorization Specification Langage) (Jajodia *et al*, 1997) est un langage formel et logique pour la spécification des politiques de contrôle d'accès. Ce langage n'est pas facilement traduisible en une implémentation de bas niveau, de plus ses constructions ne sont pas appropriées à ce domaine. L'approche utilisée dans LaSCO (Language for Security Constraints on Objects) (Hoagland *et al*, 1998), est une approche graphique de la spécification des contraintes pesant sur les objets. Le langage Mirò (Heydon *et al*, 1990), est lui aussi un langage qui peut être visuel grâce aux outils de visualisation qui l'accompagnent. Le langage PDL (Policy Description Language) (Lobo *et al*, 1999), est un langage basé sur les événements et mis au point dans les laboratoires Bell-Labs. Marriot présente et compare plusieurs de ces langages dans (Marriot *et al*, 1994). D'autres

exemples de langages de spécification, qui n'ont pas été créés dans le but précis de spécifier des politiques de sécurité mais ont été adaptés à ce besoin, sont les langages SPML (Debbabi *et al*, 2001) et Z. Le domaine d'application de toutes ces approches est très limité pour satisfaire les exigences de sécurité.

7. Conclusion et travaux futurs

Dans cet article nous avons présenté une nouvelle approche pour spécifier les politiques de sécurité. Cette approche se base sur la définition d'un nouveau langage nommé PPL. Le processus de développement de ce langage, s'est basé sur une étude et une analyse profonde du domaine des politiques de sécurité. Cette analyse, nous a permis de capturer les concepts clés de ce domaine, et ainsi d'en définir les cinq niveaux d'abstraction.

Nous avons utilisé PPL pour spécifier une variété de politiques de sécurité typiques ainsi que d'autres spécifiques à des domaines particuliers. Ces spécifications ont validé expérimentalement l'expressivité de notre langage.

Nous avons développé une première version d'un compilateur pour notre langage qui génère une implémentation d'une politique à partir de sa spécification. Cette implémentation est préliminaire, et il y a plusieurs optimisations à explorer. En particulier, nous travaillons sur l'optimisation du processus de détection de conflits et d'anomalies et sur l'instauration d'un protocole de négociation entre politiques lors d'un conflit, chose qui va permettre sa correction automatique. Nous étudions aussi l'optimisation du processus de raffinage par la détermination au niveau de la spécification des ressources nécessaires pour l'implémentation de la politique.

Ces idées sont des exemples d'optimisations spécifiques au domaine des politiques de sécurité, elles peuvent être effectuées par l'ajout de plus d'informations explicites au niveau langage puis leur factorisation dans le compilateur.

Bibliographie

Scott Thibault *" domain specific language: conception, implementation and application"* PhD thesis, University of Rennes1, France, October 1998

National Research Council. Computers at Risk : Safe Computing in the Information Age, 1991. National Academy Press**.**

.Sandhu, R.S. and P. Samarati, *Authentication, Access Control, and Intrusion Detection*. Part of the paper appeared under the title "Access Control: Principles and Practice" in IEEECommunications, 1994. 32(9): p. 40-48**.**

Abrams, M.D. *Renewed Understanding of Access Control Policies*. In Proceedings of 16th National Computer Security Conference. 1993. Baltimore, Maryland, U.S.A.

SLOMAN, M. S. Policy Driven Management for Distributed Systems. *Journal of Network and Systems Management* 2(4): 333-360**.**

N. Damianou, N. Dulay, E. Lupu et M. Sloman. The ponder policy specification language. POLICY 2001, pages 18–38, 2001.

S. Jajodia, P. Samarati et V. Subrahmanian. A logical language for expressing authorizations. Proceedings of the 1997 IEEE Symposium on Security and Privacy, pages 31–42, 1997**.**

J. Hoagland, R. Pandey et K. Levitt. Security policy specification using a graphical approach. Rapport technique CS-98-3, University of California, Dept. of Computer Science, Davis, California, July 1998.

A. Heydon, M.Maimone, J. Tygar, J.Wing et A. Zaremski. Mir`o : Visual specification of security. IEEE Transactions on Software Engineering, 16(10): 1185–1197, 1990.

J. Lobo, R. Bhatia et S. Naqvi. A policy description language. Proceedings of AAAI, pages 291–298, 1999.

D. Marriot, M. Mansouri-Samani et M. Sloman. Specification of management policies, 1994. Presenté au Fifth IFIP/IEEE International Workshop on Distributed Systems : Operations & Management (DSOM'94), Toulouse, France.

M. Debbabi, M. Girard, L. Poulin et M. Salois. Dynamic Monitoring of Malicious Activity in Software Systems. Symposium on Requirements Engineering for Information Security (SREIS'01), mars 2001. Groupe de recherche LSFM, département d'informatique, Université Laval, Canada**.**

R.J. Andreson. A security policy model for clinical information systems. In *1996 IEEE Symposium on Security and Privacy,* pages 30-40. IEEE Computer Society Press, 1996.

N.C Damianou. *Policy Framework for the Management of Distributed systems.* PhD thesis, Imperial College. London, UK, Februray 2002.

Tim Finin, Lalana Kagal et Anupam Joshi. Web services security

Bhatia R. Lobo, J. et S. Navqi. A policy description language, July 1999

Tim Finin, Lalana Kagal et Anupam Joshi. A policy based approach to security for semantic web, Octobar 2000.

Gerhard Wolfien. Network IPSEC specification. In *Pan-European Harmonisation of Vehicle Emergency call service chain.* Information society Technologie (IST). October 2002

Annexe

Spécification de la politique de sécurité des Firewalls distribués en PPL

```
rule R1 () {
subject : firewall1 ;  /* firewall du LAN1 */
object: packet;
if (protocol = TCP && direction= outgoing && src_ip = 193.95.30.1 && dst_ip
=     193.95.60.10 && src_port = any && dst_port = 53 && ) then allow; }
rule R2 () {
subject : firewall1 ;  /* firewall du LAN1 */
object: packet;
if (protocol = TCP && direction= outgoing && src_ip = 193.95.30.2 && dst_ip
=                    193.95.60.20  &&  src_port  =  any  &&  dst_port  =  25   )
then allow; }
rule R3 () {
subject : firewall1 ;  /* firewall du LAN1 */
object: packet;
if (protocol = TCP && direction= incoming && src_ip = any && dst_ip
     = (193.95.30.4 && 193.95.30.3 &&  src_port = any && dst_port = (21 ||
     80)  ) then allow; }
rule R8 () {
subject : firewall1 ;  /* firewall du LAN1 */
object: packet;
if (protocol = ICMP && direction= incoming && src_ip = any && dst_ip
     = any &&  type = 8 && code = 0  ) then deny; }
rule R10 () {
subject : firewall1 ;  /* firewall du LAN1 */
object: packet;
if (protocol = ICMP && direction= incoming && src_ip = any && dst_ip
     = any &&  type ={0,8, 9,10,13,14,15,16,17,18}  && code = any   ) then
     deny;}
rule R12 () {
subject : firewall1 ;  /* firewall du LAN1 */
```

```
object: packet;
if (protocol = ICMP && direction= outgoing && src_ip = any && dst_ip
    = any &&  type = 3 && code = 0) then accept; }
rule R14 () {
subject : firewall1 ;  /* firewall du LAN1 */
object: packet;
if (protocol = ICMP && direction= outgoing && src_ip = any && dst_ip
    = any &&  type ={0,8, 9,10,13,14,15,16,17,18}  && code = 0   ) then
    deny; }
rule R4 () {
subject : firewall2 ;  /* firewall du LAN2 */
object: packet;
if (protocol = TCP && direction= outgoing && src_ip = any && dst_ip     =
              172.16.16.X && src_port = any && dst_port = 80 )  then allow; }
rule R7 () {
subject : firewall2 ;  /* firewall du LAN2 */
object: packet;
if (protocol = ICMP && direction= outgoing && src_ip = any && dst_ip
    = (193.95.30.X && 172.16.16.X) &&  type = 8 && code = 0) then Allow;
    }
rule R91 () {
subject : firewall2 ;  /* firewall du LAN2 */
object: packet;
if (protocol = ICMP && direction= incoming && src_ip = any && dst_ip
    = 172.16.50.X &&  type = 8 && code = 0) then deny; }
rule R92 () {
subject : firewall2 ;  /* firewall du LAN2 */
object: packet;
 if (protocol = ICMP && direction= incoming && src_ip = 193.95.30.X &&
dst_ip = 172.16.50.X &&  type = 8 && code = 0) then allow; }
rule R93 () {
subject : firewall2 ;  /* firewall du LAN2 */
```

```
object: packet;
if (protocol = ICMP && src_ip = 172.16.16.X && dst_ip = 172.16.50.X &&
type = 8 && code = 0) then deny; }
rule R5 () {
subject : firewall3 ;  /* firewall du LAN3 */
object: packet;
if (protocol = TCP && direction= outgoing && src_ip = any && dst_ip =
      172.16.50.1 && src_port = any && dst_port = 21  ) then allow; }
rule R6 () {
subject : firewall3 ;  /* firewall du LAN3*/
object: packet;
if (protocol = ICMP && direction= outgoing && src_ip = any && dst_ip =
172.16.50.X &&type = 8          && code = 0) then deny; }
rule R11 () {
subject : firewall3 ;  /* firewall du LAN3*/
object: packet;
if (protocol = ICMP && direction= outgoing && src_ip = any && dst_ip = any
      &&type = {3,4,5,9,11,12}&& code = any then deny; }
rule R13 () {
subject : firewall3 ;  /* firewall du LAN3*/
object: packet;
if (protocol = ICMP && direction= outgoing && src_ip = any && dst_ip = any
      &&type = 13 && code = any then accept; }
```

```
policy UnivPolicy{
      scope = {F1; F2; F3; all packets}  //Fi  : Firewalli
      body = { R1;R2;R3;R4;R5;R6;R7;R8;R91;R92;R93; R10; R11; R12; R13;
               R14}
}
```

Etude de faisabilité concernant le transfert de contexte pour la sécurité

Fabien ALLARD

France Télécom Recherche & Développement
38-40 rue du Général Leclerc
F-92794 Issy-Les-Moulineaux
fabien.allard@francetelecom.com

RÉSUMÉ. *L'utilisation d'Internet doit pouvoir se faire en confiance pour l'utilisateur mais cette sécurité a un coût pour les FAIs. Dans un contexte de mobilité, cette sécurité doit être remise en place à chacun des mouvements des clients. Ainsi, un nouveau type de mécanisme a vu le jour dans les instances de standardisation : le transfert de contexte. Celui-ci a pour but de transmettre, d'un équipement à un autre, les données adéquates pour que les services rendus n'aient pas de discontinuités. Ce mécanisme pourrait être un atout pour un opérateur car il lui permettrait d'assurer le même niveau de sécurité pendant les déplacements des usagers dans le réseau tout en réduisant les coûts. Dans cet article, nous faisons tout d'abord un état de l'art des protocoles de transfert de contexte. Puis, nous définissons le contexte IPsec. Enfin, nous décrivons l'utilisation de CXTP pour IPsec.*

ABSTRACT. *The use of internet must be able to be in confidence for the user but this security has a cost for ISPs. In a mobility context after an handover, this security must be set up from scratch for each customer. So, a new kind of mechanism has been designed in the standardization authorities : the transfer context. The aim of this mechanism is to transfer between equipments suitable data to avoid services discontinuity. This mechanism could be an asset for an operator because it could allow same security level during the users hand-over in the network with less cost. In this article, we first do a state-of-art of transfer context protocols. Afterwards, we define the IPsec context. Finally, we describe the use of CXTP for IPsec.*

MOTS-CLÉS : *Transfert de contexte, CXTP, sécurité des réseaux, IPsec.*

KEYWORDS: *Context Transfer, CXTP, network security, IPsec.*

1. Introduction

En plus de trente ans, le réseau Internet a considérablement évolué. Bien que l'infrastructure même du réseau des réseaux gagne en puissance et en densité, un grand changement se produit aussi du côté utilisateur. En effet, avec l'apparition de la téléphonie sans-fil (GSM, GPRS, CDMA, UMTS) et l'évolution de l'informatique en général, de plus en plus d'utilisateurs possèdent des équipements légers et mobiles. D'autre part, l'accès à l'Internet sans-fil se démocratise, chaque jour de nouveaux *hot-spots* (zones d'accès sans-fil) voient le jour et les ordinateurs portables sont équipés (en série) de cartes WiFi. Les opérateurs travaillent maintenant à la mise en place d'un Internet ambiant, où l'accès au réseau IP ne se fera plus seulement depuis le domicile ou l'entreprise, mais de n'importe quel endroit et ce sans fil.

Lors de l'utilisation du réseau d'accès, les équipements se configurent pour offrir des services aux clients. En effet, la sécurité, la qualité de service, les services comme la voix sur IP nécessitent la mise en place d'informations et d'états sur de nombreux équipements. Ces états sont le résultat d'échanges de messages entre le client et le réseau d'accès reposant sur des protocoles distincts (e.g. `IKE` (Harkins *et al.*, 1998) pour IPsec). Cette signalisation peut être importante et durer un temps non négligeable. La combinaison de la mobilité et de l'usage de certains services pose cependant problème. En effet, tout le réseau de l'opérateur n'est pas configuré pour l'équipement de l'utilisateur, seuls les équipements sur lesquels une configuration est nécessaire le sont. Ainsi, dès que le mobile se déplace, il faut rétablir les contextes avec le nouvel équipement d'accès. L'importance de la signalisation mise en œuvre pour maintenir le service risque de considérablement dégrader les performances lors des déplacements. Une solution à ce problème est d'effectuer des transferts de contexte d'un équipement d'accès à un autre et d'utiliser ainsi les paramètres (le contexte) établis sur l'ancien équipement pour configurer le nouveau.

Dans cet article, nous considérons que nous sommes dans un contexte de mobilité IP et une architecture de type opérateur ou entreprise. Dans la partie 2, nous faisons un état de l'art des protocoles de transfert de contexte et nous détaillons le protocole `CXTP` de l'IETF. Dans la partie 3, nous étudions un cas d'utilisation de ce mécanisme pour la sécurité en définissant le contexte IPsec. Enfin dans la partie 4, nous décrivons l'utilisation de `CXTP` pour IPsec.

2. Les protocoles de transfert de contexte

Lorsqu'un mobile se déplace dans un réseau d'accès, il s'associe à de nouveaux équipements. L'idée est de transférer les contextes établis au niveau d'un point d'accès (niveau liaison) ou d'un routeur d'accès (niveau réseau) vers un autre (cf. Figure 1). Pour cela, il faut être capable de transférer les informations nécessaires au rétablissement d'un état dans le nouvel équipement permettant l'accès du mobile. En fin de compte, le nouvel équipement doit se retrouver dans le même état que s'il avait réalisé une initialisation complète du service avec le mobile.

Nous avons constaté que le nombre de protocoles de transfert de contexte est limité puisque seulement une solution est réellement exploitable. Néanmoins, dans cette partie nous exposons rapidement la vision de l'IEEE de ce mécanisme avec les standards `802.11f` et `802.11r` avant d'aborder la solution expérimentale de l'IETF : le `Context Transfer Protocol` (`CXTP`).

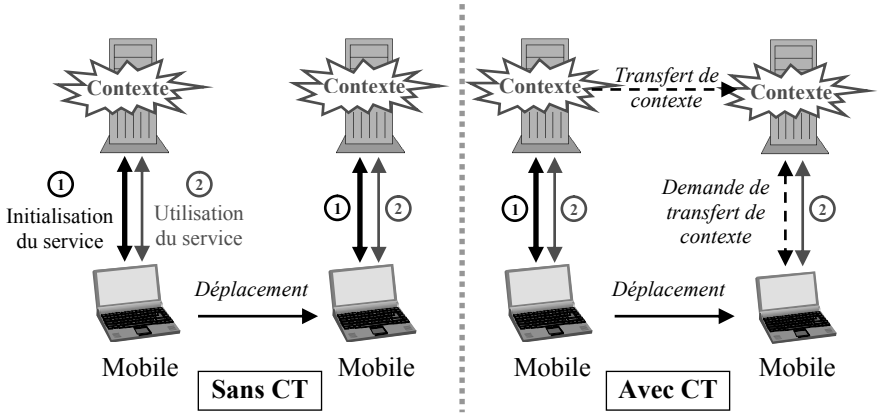


Figure 1. ***Déplacement de mobile avec et sans transfert de contexte***

2.1. *Les standards IEEE 802.11f et IEEE 802.11r*

Les protocoles de l'IEEE étant des protocoles de **niveau liaison**, il s'agit donc ici de transfert de contexte de niveau liaison entre points d'accès.

2.1.1. *Le standard `IEEE 802.11f`*

L'objectif de ce standard (802.11f, 2003) est de décrire les pratiques recommandées pour l'exécution du protocole `Inter Access Point Protocol` (`IAPP`). Ce standard définit uniquement les primitives d'accès aux données d'un contexte et la possibilité de transfert de contexte entre points d'accès (donc au niveau liaison). Ce protocole peut être utilisé pour implémenter des mécanismes de gestion rapide de la mobilité (*fast handoff*) ou pour transférer les contextes de sécurité du standard `802.11i` (Kassab *et al.*, 2005).

2.1.2. *La standard `IEEE 802.11r`*

Ce standard encore discuté en groupe de travail (*Tasking Group r*), a pour vocation d'apporter des solutions précises au problème de *fast handoff*. L'objectif de `802.11r` est de définir un protocole de gestion rapide des changements de point d'accès pour optimiser le délai entre la dernière trame de données reçue du premier point d'accès et la première du deuxième. Néanmoins, ce standard ne décrit pas comment les données sont transférées d'un point d'accès à l'autre. Il s'appuiera, a

priori, sur l'utilisation d'un protocole de transfert de clés en cours de définition à l'IETF. Le draft de ce groupe de travail devrait être finalisé d'ici fin 2006.

2.2. *Le RFC 4067 – `Context Transfer Protocol`*

Dans cette section, nous présentons en détail le protocole de transfert de contexte défini originellement par le groupe de travail *Seamoby* (*Seamless Mobility*) de l'IETF (Kempf, 2002). Le résultat est le RFC 4067 « `Context Transfer Protocol (CXTP)` » (Loughney *et al.*, 2005).

2.2.1. *Le principe*

Le protocole `CXTP` est un protocole de **niveau réseau** réalisant des transferts de contexte entre *routeurs d'accès* (AR). Il est générique dans le sens où il transfère n'importe quel contexte : les applications souhaitant l'utiliser doivent définir leur contexte et obtenir un identifiant (cf. *CXTP for PANA* (Bournelle *et al.*, 2006)). Ce dernier est un numéro qui va permettre à l'ancien routeur d'accès (pAR) d'identifier le contexte à transférer vers le nouveau routeur d'accès (nAR). Le transfert de contexte peut être initié soit par le mobile (*mobile controlled*), soit par le réseau (*netword controlled*).

2.2.2. *Les messages*

Le protocole `CXTP` définit trois types de messages pour réaliser le transfert. Le nouveau routeur demande les contextes à transférer par le message `Context Transfer Request` (`CT-Request`). Il contient l'ancienne adresse IP du mobile et un jeton d'autorisation. L'adresse IP permet au routeur de retrouver le contexte tandis que le jeton est utilisé pour autoriser le transfert. Si le jeton est valide, le précédent routeur envoie le message `Context Transfer Data` (`CTD`) qui contient les contextes et éventuellement du matériel cryptographique. Celui-ci sera employé par le nouveau routeur d'accès pour valider la demande d'activation du contexte transféré (envoyée par le mobile). Le nouveau routeur indique le résultat de l'opération grâce au message `Context Transfer Reply` (`CT-Reply`) (cf. Figure 2).

Pour déclencher et activer le transfert, le mobile utilise le message `Context Transfer Activate Request` (`CTAR`). Ce message est envoyé au nouveau routeur et si possible à l'ancien avant de réaliser le déplacement. Le routeur a la possibilité d'indiquer au mobile si le transfert s'est correctement effectué en utilisant le message `Context Transfer Activate Acknowledgement` (`CTAA`).

Les messages sont utilisés suivant deux modes de fonctionnement : prédictif ou réactif.

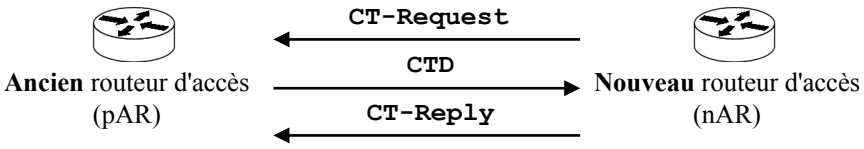


Figure 2. *Transfert de contexte entre routeurs d'accès*

2.2.3. *Le mode prédictif ou anticipé*

Dans ce mode, le mobile (MN) sait vers quel routeur il va se déplacer et anticipe le transfert. Pour cela, il envoie un message `CTAR` à son routeur d'accès (cf. Figure 3) contenant l'adresse du nouveau routeur, les contextes à transférer et un jeton d'autorisation. Ce dernier est calculé sur l'ensemble du message et suppose que le mobile et le routeur d'accès précédent disposent d'une clé partagée (le RFC 4067 ne précise pas comment cette clé est obtenue).

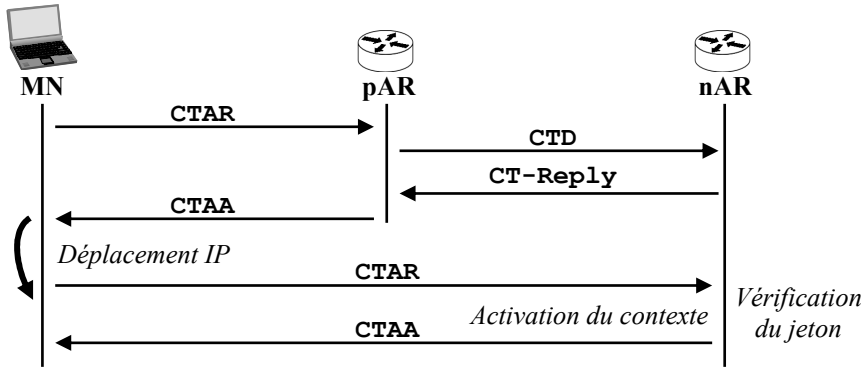


Figure 3. *Mode prédictif de CXTP*

Le routeur vérifie le jeton, s'assure qu'il possède bien les contextes demandés et les envoie dans le message `CTD` au routeur indiqué avec du matériel cryptographique. Le nouveau routeur met en place le contexte mais ne l'active pas encore. Lorsque le mobile se déplace, il envoie un message `CTAR`. Ce dernier contient aussi un jeton et c'est grâce au matériel cryptographique envoyé par le précédent routeur que le nouveau vérifie la validité du jeton. Si elle est correcte, le contexte est activé.

2.2.4. *Le mode réactif*

Dans ce mode, le mobile s'est déjà déplacé avant de demander le transfert. Il envoie au nouveau routeur d'accès un message `CTAR` (cf. Figure 4) contenant cette

fois-ci l'adresse de son ancien routeur, son ancienne adresse IP ainsi qu'un jeton d'autorisation. Le mobile et le nouveau routeur ne se connaissant pas, c'est l'ancien routeur qui va valider le transfert. Le nouveau routeur envoie un message `CT-Request` contenant les données précédentes à l'adresse indiquée. L'ancien routeur valide la demande en vérifiant le jeton puis transfère les contextes dans le message `CTD`. Enfin, le nouveau routeur met en place les contextes et informe éventuellement le mobile du résultat par le message `CTAA`.

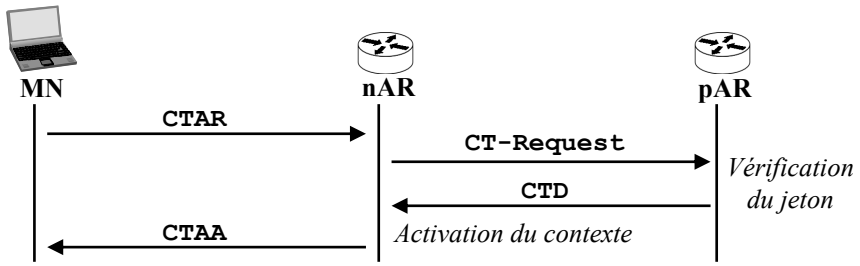


Figure 4. ***Mode réactif de CXTP***

2.2.5. *Transport et sécurité entre routeurs d'accès*

Les messages échangés entre routeurs d'accès sont protégés par l'utilisation du protocole `Encapsulating Security Payload` (`ESP`) (Kent *et al.* (ESP), 1998). Afin de faciliter la tâche de l'administrateur, le protocole `Internet Key Exchange` (`IKE`) peut être utilisé pour mettre en place et maintenir les associations de sécurité IPsec. Cependant ces associations doivent être mises en place **avant** le transfert d'un contexte. En effet, tout le bénéfice des transferts de contexte anticipés serait perdu s'il était nécessaire d'initier une nouvelle association de sécurité IPsec (Kent *et al.* (IPsec), 1998) avant l'échange `CXTP`. Le choix du protocole de transport (`TCP`, `UDP`, `SCTP`) est encore à l'étude par le groupe de recherche *Mobopts* [1].

3. Le contexte IPsec

Dans cette partie, nous définissons le contexte IPsec. Nous considérons le cas où les associations de sécurité (SA) IPsec sont établies entre un mobile et un routeur d'accès (différent du cas de SAs établies entre deux ARs). Lorsque le mobile se déplace, il faut donc établir à nouveau ces associations de sécurité avec le nouveau routeur d'accès. L'idée est de transférer le contexte IPsec de l'ancien routeur d'accès (pAR) vers le nouveau (nAR).

1. Groupe de recherche formé à l'IRTF (*Internet Research Task Force*) afin de mener des recherches longues sur les optimisations de la mobilité IP.

Lorsque deux hôtes partagent une SA IPsec, ils font appels à deux bases de données :

- **la SAD** (*Security Association Database*) : Elle contient tous les paramètres relatifs à chaque SA et est consultée pour savoir comment traiter chaque paquet reçu ou à émettre,
- **la SPD** (*Security Policy Database*) : Elle est établie et maintenue par un utilisateur, un administrateur ou une application et permet de décider, pour chaque paquet, s'il se verra apporter des services de sécurité, s'il sera autorisé à passer outre ou s'il sera rejeté.

Une association de sécurité peut être mise en place soit de manière manuelle, soit de manière automatique en utilisant `IKE`.

3.1. *Associations de sécurité IPsec mises en place manuellement*

Dans ce cas, on suppose que les SAs sont mises en place sans l'aide de `IKE`. Il faut donc déplacer les données relatives à ces SAs contenues dans la SAD et la SPD du pAR vers le nAR. Il faudra ensuite mettre à jour ces deux bases sur le MN et les configurer sur le nAR.

3.1.1. *Le contexte SAD*

Dans la SAD, chaque SA est identifiée à l'aide des sélecteurs suivants :

- l'**adresse IP destination** des paquets,
- l'**identifiant du protocole de sécurité** c'est-à-dire `AH` (Kent *et al.* (AH), 1998) ou `ESP`,
- le **SPI** (*Security Parameter Index*). Il permet de distinguer les différentes SAs pour une même destination utilisant le même protocole IPsec (`AH` ou `ESP`). Il est choisi par le récepteur.

De plus, dans le cas de communications bidirectionnelles, nous avons deux entrées dans la SAD (entrante et sortante) pour chacune d'entre elles. Chaque entrée est définie par les champs suivants :

- le **compteur de numéro de séquence** : Valeur utilisée pour générer le champ du numéro de séquence des en-têtes `AH` ou `ESP`,
- la **fenêtre d'anti-rejeu** : Valeur utilisée pour déterminer si un paquet `AH` ou `ESP` entrant est rejoué,
- l'**algorithme d'authentification `AH` et les clés**,
- les **algorithmes d'authentification** et **de chiffrement `ESP` et les clés**,
- le **temps de vie de la session**,
- le **mode IPsec** : Transport ou Tunnel,
- le **Path MTU** qui indique la taille maximale des paquets IPsec.

Les sélecteurs et champs ci-dessus constituent le contexte SAD qui sera transféré d'un routeur d'accès à un autre.

Après que le MN se soit déplacé, il faut que sa SAD soit mise à jour et que la SAD du nAR soit configurée. Les modifications à apporter respectivement sont présentées par le Tableau 1 et le Tableau 2.

	@IP destination	Proto.	SPI	Entrées
Ancienne SA	ancienne @IP MN	ESP	spi-mn-in	SA1
Nouvelle SA	**nouvelle @IP MN**	ESP	spi-mn-in	SA1
Ancienne SA	@IP pAR	ESP	spi-mn-out-1	SA2
Nouvelle SA	**@IP nAR**	ESP	**spi-mn-out-2**	SA2

Tableau 1. *Security Association Database du MN*

	@IP destination	Proto.	SPI	Entrées
SA entrante	@IP nAR	ESP	spi-mn-in	SA2
SA sortante	@IP MN	ESP	spi-mn-out-2	SA2

Tableau 2. *Security Association Database du nAR*

Plusieurs points peuvent cependant poser problème :

- Il peut arriver une **collision du SPI** lorsque le contexte SAD du pAR est transféré au nAR. En effet, il se peut que le SPI utilisé par le pAR (spi-mn-out-1) soit déjà utilisé par le nAR. Dans ce cas, il est nécessaire de le modifier (d'où spi-mn-out-2) et d'en informer le MN.
- Lors du déplacement IP du MN, il n'est pas impossible que des paquets en provenance du MN vers le pAR se perdent, ce qui conduit à une **incohérence entre le compteur de numéro de séquence transféré (celui du pAR) et le compteur réel**. Néanmoins, ce problème n'existe que si la différence entre le compteur réel et le compteur transféré est supérieure à la fenêtre d'anti-rejeu.

3.1.2. *Le contexte SPD*

Dans la SPD, les sélecteurs utilisés pour appliquer la politique de sécurité sont :

- l'**adresse IP destination** des paquets (MN ou AR),
- l'**adresse IP source** des paquets (MN ou AR),
- le **protocole de niveau supérieur** qui doit être protégé.

En plus de ceux-ci, il sera nécessaire de transférer la politique de sécurité associée aux SAs devant être déplacées.

Les sélecteurs et la politique de sécurité constituent le contexte SPD qui sera transféré d'un routeur d'accès à un autre.

Après que le MN se soit déplacé, il faut que sa SPD soit mise à jour et que la SPD du nAR soit configurée. Les modifications à apporter respectivement sont présentées par le Tableau 3 et le Tableau 4.

	@IP destination	@IP source	Proto. Sup.
Ancienne SP entrante	ancienne @IP MN	@IP pAR	n° proto.
Nouvelle SP entrante	**nouvelle @IP MN**	**@IP nAR**	n° proto.
Ancienne SP sortante	@IP pAR	ancienne @IP MN	n° proto.
Nouvelle SP sortante	**@IP nAR**	**nouvelle @IP MN**	n° proto.

Tableau 3. *Security Policy Database du MN*

	@IP destination	@IP source	Proto. Sup.
SP entrante	@IP nAR	nouvelle @IP MN	n° proto.
SP sortante	nouvelle @IP MN	@IP nAR	n° proto.

Tableau 4. *Security Policy Database du nAR*

On pourra noter que généralement l'opérateur met en place une politique de sécurité (donc une SPD) identique sur tous ses routeurs d'accès. Le contexte SPD n'aura donc pas forcément besoin d'être transféré.

3.2. *Associations de sécurité IPsec mises en place dynamiquement à l'aide de IKE*

L'objectif est de déplacer le contexte IPsec sans avoir à relancer `IKE` après le déplacement. Comme pour le cas où les SAs sont mises en place manuellement, il faut déplacer les données relatives à celles-ci contenues dans la SPD du pAR vers le nAR, les configurer sur le nAR et les mettre à jour sur le MN (cf. paragraphe 3.1.2). De plus, il faut également déplacer les données créées par `IKE` pour ces SAs.

3.2.1. *Le contexte IKE*

L'échange `IKE` est divisé en deux phases :

Phase 1 : Négociation des associations de sécurité `ISAKMP` (Maughan *et al.*, 1998), c'est-à-dire des attributs suivants :

- l'**algorithme de chiffrement**,
- la **fonction de hachage**,
- la **méthode d'authentification**,
- le **groupe mathématique** pour les calculs relatifs à Diffie-Hellman,

et des clés permettant de protéger les messages de la phase 2 :

- **`SKEYID_e`** permettant la confidentialité des messages,

- **SKEYID_a** permettant l'authentification des messages,
- **SKEYID_d** utilisée pour la dérivation des clés utilisées par les SAs IPsec.

Phase 2 : Négociation des associations de sécurité IPsec. Les clés $KEYMAT_i$ (i étant le nombre de SAs IPsec négociées) dérivées de la clé SKEYID_d sont utilisées par les SAs IPsec pour authentifier (avec AH) ou chiffrer (avec ESP) le trafic entre le mobile et le routeur d'accès.

On pourrait se contenter de transférer le résultat de la phase 2, c'est-à-dire uniquement les paramètres caractérisant les SAs IPsec. Dans ce cas, cela reviendrait à déplacer le contexte SAD comme expliqué au paragraphe 3.1.1. Néanmoins, après le déplacement du MN, il serait impossible de relancer IKE pour rafraîchir les clés de la phase 2 par exemple. Il est donc intéressant de transférer également le résultat de la phase 1, c'est-à-dire les SAs ISAKMP.

Les autres données à transférer également sont :

- le *cookie* de l'*Initiator* et celui du *Responder* qui permettent d'identifier une SA ISAKMP,
- le secret partagé (IKE-PSK) utilisé pour dériver les clés de la phase 1.

Toutes ces données constituent le contexte IKE. Le contexte IPsec est donc composé des données de la SAD, de la SPD et des informations obtenues de la phase 1 de IKE.

4. Exemple de transfert de contexte IPsec avec CXTP

Dans cette partie, nous expliquons comment transférer le contexte IPsec avec le protocole CXTP défini au paragraphe 2.2. Typiquement, nous sommes dans ce cas lorsqu'un mobile WiFi souhaite monter un VPN IPsec d'entreprise et qu'il se déplace d'un réseau IP à un autre. Nous montrons ensuite quel est le gain obtenu avec cette solution.

Suite à un échange IKE entre le MN et le pAR, le MN partage des associations de sécurité IPsec avec le pAR. Il en résulte ainsi généralement un tunnel IPsec ou plus rarement un tunnel L2TP protégé par IPsec en mode transport entre le MN et le pAR. Après que le MN se soit déplacé, il envoie un message CTAR à son nouveau routeur pour lui indiquer qu'il souhaite effectuer un transfert de contexte. Le nAR demande au pAR de lui transférer le contexte IPsec défini au paragraphe 3.2 par le message CT-Req. Comme nous l'avons vu, ce contexte est composé des contextes IKE, SAD et éventuellement SPD. Le pAR envoie alors au nAR ces contextes dans le message CTD. Après configuration de ceux-ci sur le nAR et mise à jour sur le MN, le tunnel IPsec est maintenant en place entre le MN et le nAR. Il est possible de relancer IKE (phase 1 et phase 2) ultérieurement pour rafraîchir les clés des SAs ISAKMP et IPsec. La phase 2 de IKE peut être également relancée seule pour ne rafraîchir que les clés des SAs IPsec. Ces échanges sont illustrés sur la Figure 5.

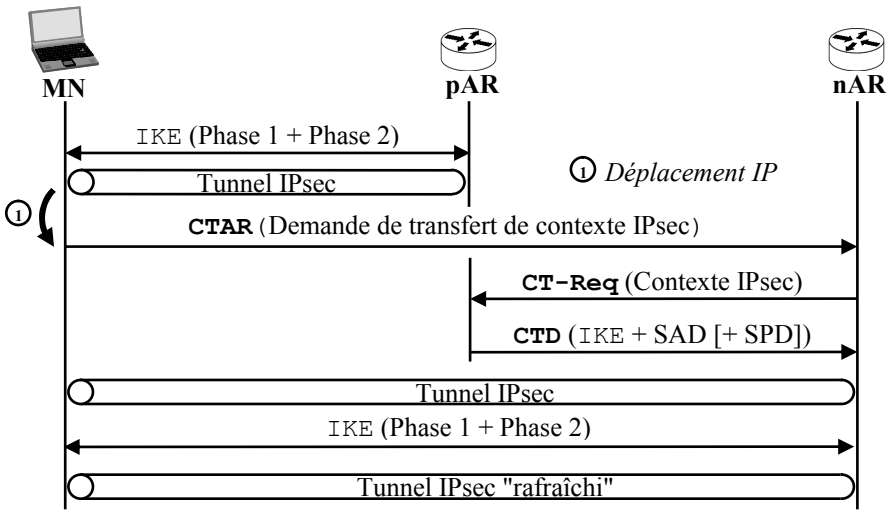


Figure 5. *Transfert de contexte en mode réactif*

Le gain du transfert de contexte pour IPsec se situe à plusieurs niveaux :

- Le nombre de messages émis (trois exactement) pour remettre en place le tunnel IPsec est moindre. En effet, en temps normal, il aurait fallu au minimum six messages (en mode agressif) pour aboutir au même résultat.
- Les opérations de génération des clés de sessions `ISAKMP` ou IPsec sont coûteuses en temps, alors qu'avec notre solution nous réutilisons les clés existantes (rien ne nous empêche de les rafraîchir après le déplacement du MN).

5. Conclusion

Dans cet article, nous avons présenté le mécanisme de transfert de contexte et réalisé un état de l'art des solutions existantes. Notre intérêt pour ce mécanisme se situe au niveau des améliorations de performances qu'il peut apporter au niveau de la mobilité IP tout en garantissant un niveau de sécurité inchangé. C'est pour cette raison que nous avons essayé de le mettre en pratique pour IPsec, tout d'abord en définissant le contexte IPsec, puis en expliquant l'utilisation de `CXTP` pour IPsec. Nous avons réalisé une implémentation en langage C pour en valider la viabilité avec le transfert d'un compteur. Cependant cette implémentation ne nous permet pas à l'heure actuelle d'être appliquée à des implémentations courantes de protocoles de sécurité. Une prochaine étape pour amener notre solution à terme sera l'étude du contexte `IKEv2` (Kaufman, 2005) et l'intégration de cette solution dans l'architecture `PANA` (Forsberg *et al.*, 2005). Enfin, nous simulerons ce mécanisme pour mesurer les gains en termes de performances lors des déplacements mais aussi

le comparer à d'autres solutions comme par exemple la pré-authentification (Dutta *et al.*, 2006).

6. Remerciements

Je tiens à remercier Jean-Michel Combes (France Télécom R&D), Jean-Marie Bonnin (GET/ENST Bretagne) et Julien Bournelle (GET/INT) pour leur soutien et l'aide qu'ils m'ont apportée durant la rédaction de cet article.

7. Bibliographie

Bournelle J., Laurent-Maknavicius M., Tschofenig H., El Mghazli Y., Giaretta G., Lopez R., Ohba Y., "Use of Context Transfer Protocol (CXTP) for PANA", draft-ietf-pana-cxtp-01, IETF, 2006.

Bournelle J., "Vers un système d'authentification intégrant la configuration dynamique de la mobilité IPv6 et la prise en compte des déplacements", Thèse de doctorat, INT, 2004.

Dutta A., Fajardo V., Ohba Y., Taniuchi K., Schulzrinne H., "A Framework of Media-Independent Pre-Authentication (MPA)", draft-ohba-mobopts-mpa-framework-02, IETF, 2006.

Forsberg D., Ohba Y., Patil B., Tschofenig H., Yegin A., "Protocol for Carrying Authentication for Network Access (PANA)", draft-ietf-pana-pana-10, IETF, 2005.

Harkins D., Carrel. D., "The Internet Key Exchange (IKE)", RFC 2409, IETF, 1998.

Kaufman C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, IETF, 2005.

Kassab M., Belghith A., Bonnin J-M., Sassi. S., "Fast Pre-Authentication Based on Proactive Key Distribution for 802.11 Infrastructure Networks", WMuNeP 2005, Montreal, 2005.

Kempf J., "Problem Description: Reasons For Performing Context Transfers Between Nodes in an IP Access Network", IETF, RFC 3374, 2002.

Kent S., Atkinson R., "Security Architecture for the Internet Protocol", RFC 2401, IETF, 1998.

Kent S., Atkinson R., "IP Authentication Header (AH)", RFC 2402, IETF, 1998.

Kent S., Atkinson R., "IP Encapsulating Security Payload (ESP)", RFC 2406, IETF, 1998.

Loughney J., Nakhjiri M., Perkins C., Koodli R., "Context Transfer Protocol (CXTP)", RFC 4067, IETF, 2005.

Maughan D., Schertler M., Schneider M., Turner J., " Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2408, IETF, 1998.

802.11f, "IEEE Trial-Use Recommended Practice for Multi-Vendor Access Point Interoperability via an Inter-Access Point Protocol Across Distribution Systems Supporting IEEE 802.11 Operation", IEEE, IEEE 802.11f, 2003.

Infrastructure pour la gestion de politiques de sécurité IPSec

M. El Hamzaoui

Université Hassan II – Aïn Chock
Faculté des Sciences
B.P 5366, Maârif – Casablanca. Maroc
m_elhamzaoui@yahoo.fr

A. Sekkaki, B. Bensassi

Université Hassan II – Aïn Chock
Faculté des Sciences
B.P 5366, Maârif – Casablanca. Maroc
a.sekkaki@fsac.ac.ma , bahloul_bensassi@yahoo.fr

RÉSUMÉ *Vu les exigences incessantes des clients, la grande extension des environnements de la gestion des systèmes distribués et les modifications permanentes qu'ils subissent, etc.., les environnements de la gestion de la sécurité des communications inter-domaines doivent être généralement dynamiques et gérables à base de politique.*

La gestion à base de politique exige la détermination de l'ensemble des stratégies et tactiques déterminant la politique de sécurité à suivre. Pour distribuer et appliquer cette politique, il faut la traduire en un ensemble de règles. Ensuite, ces règles doivent être communiquées à des équipements de décision (PDP: Policy Decision Point) et à d'autres équipements d'application de politique (PEP: Policy Enforcement Point) étendus sur le réseau.

Dans ce travail, nous allons présenter une infrastructure dynamique pour la gestion de la sécurité IPSec des communications inter-domaines. Cette infrastructure sera composée d'un Serveur de Sécurité IPSec (IPSecServ) et d'un Système de Surveillance (MS : Monitoring Service) pour automatiser le fonctionnement de l'IPSecServ. Pour la spécification de nos politiques de sécurité, on utilisera le langage Ponder qui apparaît actuellement le meilleur outil de spécification des politiques de sécurité et de gestion pour les systèmes distribués. Un prototype est implémenté moyennant l'environnement CORBA et quelques résultats expérimentales seront également présentés.

MOTS-CLÉS : *Domaine, Gestion à base de politique, PDP, PEP, Langage Ponder.*

1. Introduction

Vu les problèmes de la gestion et le grand nombre d'entreprises que doit gérer un environnement de la gestion des communications inter-domaines, la gestion de la sécurité de communications inter-domaines doit être dynamique et à base de politique. Dans ce contexte, le langage Ponder (Damianou et al., 2001) est très utile pour spécifier les politiques de sécurité et celles de gestion pour les systèmes répartis.

Ce travail fait partie de l'ensemble des travaux de la gestion de sécurité (El Hamzaoui et al., 2005a) (El Hamzaoui et al., 2005c) , de facturation (Westphall et al., 2003) et des communications inter-domaines (El Hamzaoui et al., 2005b) réalisés au sein de notre groupe.

Nous présentons, à travers ce travail, une infrastructure dynamique pour la gestion des politiques de sécurité IPSec assurant la sécurité des communications inter-domaines. Les éléments de base de cette infrastructure seront : Le protocole IPSec (Kent et al., 1998c), Le Langage Ponder et les domaines (Sloman et al., 1994).

Un domaine est similaire à un répertoire ou à un annuaire sur un PC et il est utilisé pour partitionner les larges systèmes selon des critères bien précis (Sloman et al., 1994). Les domaines rendent la gestion des systèmes distribués facile et flexible en permettant d'apporter des modifications aux composants des domaines sans toucher aux politiques de gestion.

Le protocole IPsec (Kent et al., 1998c) offre les mécanismes nécessaires pour construire des réseaux privés virtuels (VPNs : Virtual Private Networks) qui permettent de créer, quelque soit le réseau utilisé, des tunnels sécurisés entre les communicants. Les services de sécurité assurés par IPSec sont fournis, en mode transport ou en mode tunnel, grâce aux extensions AH (Authentication Header) (Kent et al., 1998a) et ESP (Encapsulating Security Payload) (Kent et al., 1998b).

IPSec emploie les associations de sécurité (SA : Security Association) (hsc, 2002) pour faciliter la gestion des paramètres utilisés par ses extensions (les algorithmes, les clés etc.). Chaque SA est identifiée d'une façon unique à l'aide de trois paramètres: l'adresse de destination des paquets, l'identifiant du protocole de sécurité utilisé (ESP ou AH), et l'index des paramètres de sécurité (SPI : Security Parameter Index). Une SA est unidirectionnelle; par conséquent, la protection des deux sens d'une communication classique exige deux SAs, une dans chaque sens.

Ce travail sera présenté de la façon suivante; dans la deuxième section nous présenterons brièvement le langage de spécification des politiques Ponder. Notre approche de la solution sera détaillée dans la troisième section tandis que son évaluation sera présentée dans la quatrième section. Enfin, la conclusion et la perspective de ce travail feront l'objet de la dernière section.

2. Langage de spécification des politiques ‘‘Ponder’’

Ponder (Damianou et al., 2001) est un langage orienté-objet déclarative pour spécifier les politiques de sécurité et les politiques de gestion des systèmes distribués. Comme tout autre langage orienté objet, Ponder permet de déclarer des types d’objets et de les instancier pour pouvoir les réutiliser après en leur transmettant les paramètres nécessaires.

Les caractéristiques de base du langage Ponder sont :

- La spécification de contrôle d’accès qui se base sur le déploiement des politiques d’autorisation, de filtration d’information, de délégation et de freinage ;

- La spécification des politiques d’obligation pour faire appel aux agents de gestion d’intervenir lorsqu’un événement spécial a eu lieu dans le système;

- La spécification des contraintes qui constituent l’ensemble des conditions sous lesquelles certaines politiques d’obligation sont valides ;

- La spécification des politiques composées telles que les groupes, les rôles, les relations, les structures de gestion etc… pour faciliter la gestion des larges et des complexes entreprises.

Dans les travaux de la gestion à base de politique réalisés au sein d’Imperial College (Imperial College, 2003) , le PDP est appelé Sujet tandis que le PEP est appelé Cible. Pour Ponder les sujets, les cibles et les politiques sont tous organisés en domaines. L’organisation en domaines de notre environnement de la gestion de sécurité est illustrée de la manière suivante:

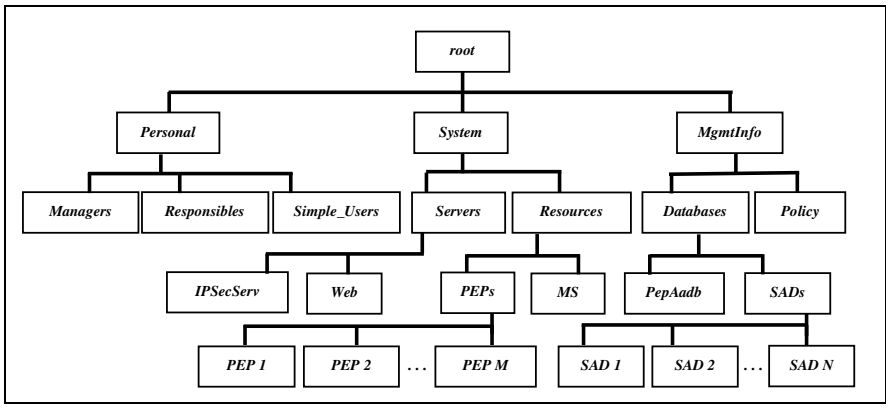


Figure 1. *Organisation en domaines des composants de notre environnement de gestion de la sécurité IPSec des communications inter-domaines.*

La gestion dynamique des composants de notre environnement de la gestion de sécurité tels que les utilisateurs, les matériels, les logiciels, etc. exige, comme il est schématisé sur la figure.1, l'emploi d'un domaine racine (root). Ce domaine racine permettra d'organiser ces composants en trois principaux sous-domaines: personal, System et MgmtInfo.

Dans ce qui suit, on se limitera aux politiques d'obligation. Une politique d'obligation détermine les actions que doivent effectuer les membres d'un seul ou de plusieurs domaines sujet, dans un système, sur les objets d'un ou de plusieurs domaines cible suite à la réception d'un événement interne ou externe qui peut être simple ou composé. Deux syntaxes de base ont été désignées pour spécifier une politique d'obligation :

- Syntaxe de déclaration directe d'une instance d'une politique d'obligation:

```
inst oblig policyName "{"
        on                      event-specification ;
        Subject     [<type>]    domain-Scope-Expression ;
        [ Target    [<type>]    domain-Scope-Expression ; ]
        do                      obligation-action-list ;
        [ catch                 exception-specification ; ]
        [ when                  constraint-Expression ; ]   ''}''
```

Le mot clé '**on**' spécifie l'événement déclenchant la politique. Le sujet (PDP) et la cible (PEP) sont exprimés en terme de domaine. Le mot clé 'catch' spécifie l'exception à exécuter en cas de problème au niveau de l'exécution des actions de la politique.

- Syntaxe de déclaration et d'instanciation d'un type de politique d'obligation:

```
Type oblig policyType ''('' formalParameters '')'' "{" {obligationation–policy-parts } ''}''
inst oblig policyName= policyType ''('' actualParameters '')'' ;
```

On déclare d'abord le type de la politique d'obligation et on l'instancie après en lui transmettant les paramètres nécessaires.

Ponder est employé dans plusieurs travaux de recherches. Ainsi, Lymberopoulos et al. (Lymberopoulos et al., 2004) ont montré comment des politiques Ponder peuvent être mises en application et validées pour des services différenciés (DiffServ) en employant CIM (Common Information Model) comme un environnement modélisant des ressources de réseau. De plus, les mêmes auteurs ont utilisé le langage Ponder pour réaliser une adaptation dynamique des politiques aux changements que peuvent avoir l'environnement géré (Lymberopoulos et al., 2003). Et enfin, Damianou et al. (Damianou et al., 2002) ont implémenté un toolkit intégré

pour la spécification, le déploiement et la gestion des politiques spécifiées par le langage Ponder.

Comme on l'a déjà mentionné, notre approche a pour objectif la gestion de la sécurité des communications inter-domaines et plus exactement, elle vise à la rendre plus dynamique. La gestion dynamique devient aujourd'hui une grande nécessité pour les grandes entreprises car les solutions classiques reposant sur le déplacement physique des administrateurs sont maintenant insuffisantes et dépassées. Notre environnement de la gestion, quant à lui, reposera sur des notions importantes telles que la gestion à base de politique intégrant ainsi des agents intelligents, le langage Ponder et les domaines.

3. Notre approche

3.1. *Principe de notre approche*

L'objectif de notre travail est la sécurisation à base de politique des communications inter-domaines. Ainsi, on s'intéressera plus aux PEPs se chargeant de la sécurité de ces communications :

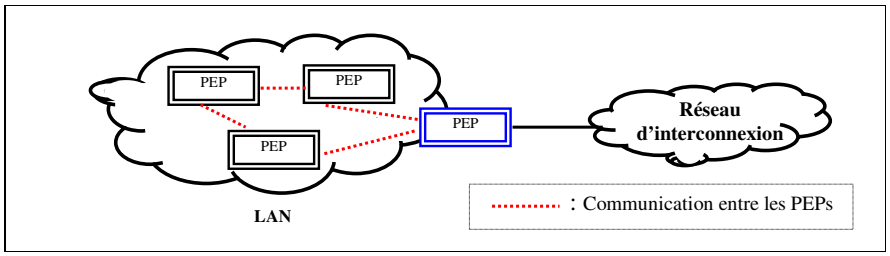


Figure 2. *Différents PEPs d'un réseau LAN de notre environnement.*

La gestion à base de politique de la sécurité des communications inter-domaines exige d'une part, l'utilisation des domaines (paragraphe 2) pour pouvoir organiser et faciliter la gestion de ces PEPs et d'autre part, l'emploi d'une plate-forme pour automatiser la gestion. Le schéma suivant montre les interactions qui pourraient avoir lieu entre les composants de notre environnement de la gestion de sécurité des communications inter-domaines :

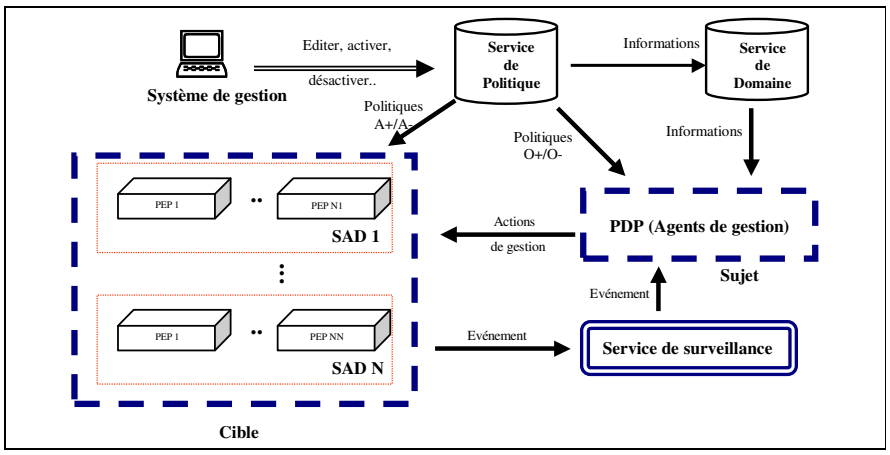


Figure 3. *Les différents composants de notre environnement de gestion de la sécurité et les interactions possibles qui pourraient être entre eux.*

Le PDP décide et distribue, en consultant le service de politique et celui de domaine, les politiques de sécurité IPSec à appliquer au niveau de ses PEPs. Un système de surveillance sera intégré également dans cette infrastructure de la gestion de sécurité afin d'automatiser son fonctionnement.

Le PDP est muni d'un ensemble de base de données des SAs (SAD : Security Association Database) (fig.1). Chaque SAD comporte trois tables SA_param, Networks_Info et PEPs_Needs. La table SA_param contient dix SAs (SA_id, Extension IPSec, algorithme, clé de cryptage, mode IPSec, SPI) tandis que la table Networks_Info contient les adresses IP et les masques réseaux nécessaires pour chaque SA. La table PEPs_Needs, quant à elle, contient la SA décidée et sélectionnée par le PDP afin de la mettre en disposition de ses PEPs.

3.2. *Fonctionnement de notre environnement de gestion de sécurité IPSec*

Pour ne pas donner la chance aux autres de découvrir nos paramètres de sécurité, on a estimé un temps d'application (T_{SA}) pour les SAs et également un temps de modification des contenus des SADs (T_{SAD} = 5* T_{SA}). Par conséquent, les contenus des SADs seront changés pendant chaque cinq modifications des SAs au niveau des PEPS. D'une part, le MS demande, pendant chaque T_{SA}, au PDP de changer la SA appliquée au niveau de ses PEPs et d'autre part, il demande, pendant chaque T_{SAD}, au PDP de changer les contenus de ses SADs. Ces deux opérations seront réalisées à

travers l'invocation des méthodes de l'interface PkiServ_MS du fichier idl (*IPSec_PKI.idl*) suivant :

```
module IPSEC_Serv {
 // Interface des méthodes invoquées par le Système de Surveillance (MS):
  interface IPSecServ_MS {
        oneway void modifySADParam();
        oneway void changeSA();
  };
 ……. // autres interfaces.
};
```

La spécification Ponder de la politique d'obligation permettant de changer les SAs appliquées au niveau des PEPs est la suivante :

```
inst oblig  PolicyMgmtSA {
 on         EventModifyAppliedSA() ;
 Subject    System/Resources/Servers/IPSecServ ;
 Target     t = MgmtInfo/Databases/SADs ;
 do         id_SA = selectSAID() –> param_SA[] = selectParam(id_SA) –> StoreSA(t.PEPs_Needs,param_SA[]);}
```

A la réception de l'événement *EventModifyAppliedSA()*, le sujet IPSecServ sélectionne aléatoirement au niveau de la table SA_param de chacune des bases de données du domaine cible (SADs) (figure1) l'identifiant d'un SA et les paramètres qui lui correspondent. Ensuite, le Sujet fait appel à la méthode *StoreSA()*. Cette méthode permet de chercher, à partir de la table Networks_Info, les informations d'adressage IP nécessaires pour terminer la composition des SAs. Ensuite, elle stocke ces SAs dans la table PEPs_Needs de la même base de données afin de les mettre à la disposition de ses PEPs.

La spécification Ponder de la politique d'obligation permettant de renouveler les contenus des SADs est :

```
inst oblig  PolicyMgmtSAD {
 on         EventModifySADs() ;
 Subject    System/Resources/Servers/IPSecServ ;
 Target     t = MgmtInfo/Databases/ SADs ;
 do         delete(t) –> prtcl=Selectpro() –> algo = SelectAlgo(prtcl)
            –> key= calculKey(prtcl, algo) –> mode = SelectMode()–> spi = CalculSPI()
            –> StoreSAD(t.SA_param, prtcl, algo, key, mode, spi) ;  }
```

A la réception de l'événement *EventModifySADs()*, le sujet IPSecServ procède de la façon suivante pour enregistrer dix nouveaux enregistrements dans les tables SA_param de chacune des bases de données du domaine cible (SADs). Tout

d'abord, le sujet fait appel à la fonction *delete()* pour écraser les contenus de toutes les tables SA_param. Ensuite, il invoque les méthodes de sélection et de calcul des paramètres de sécurité IPSec pour remplir ces tables. Enfin, les tables SA_param de chaque base de données recevront dix nouveaux enregistrements moyennant la méthode *StoreSAD()*.

Les PEPs consultent périodiquement, pendant chaque T ($T<T_{SA}$), le PDP pour lui demander la SA à appliquer et c'est la méthode *getSAParam()* de l'interface *PkiServ_PEPs* du même fichier idl (*IPSec_PKI.idl*) qui permet de réaliser cette opération.

```
module IPSEC_Serv {
    ……
  // Interface de la  méthode invoquée par les PEPs:
    interface IPSecServ_PEPs {
            string getSAParam(in string pep_id, in string pep_passwd, SAD_id) ;
    };
};
```

3.3. *Résultats de l'exécution*

Dans notre exemple, les adresses IP des différents sous réseaux (fig.4) sont de la forme 11.0.j.0 (1<j<255) tandis que l'adresse du réseau d'interconnexion est 50.0.0.0. L'adressage IP des PEPs chargés de la sécurité des communications inter-domaines est illustré de la façon suivante:

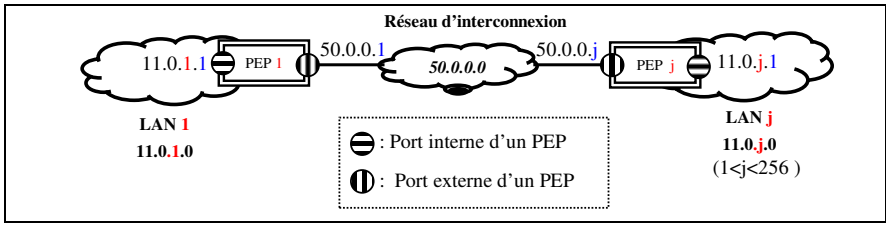
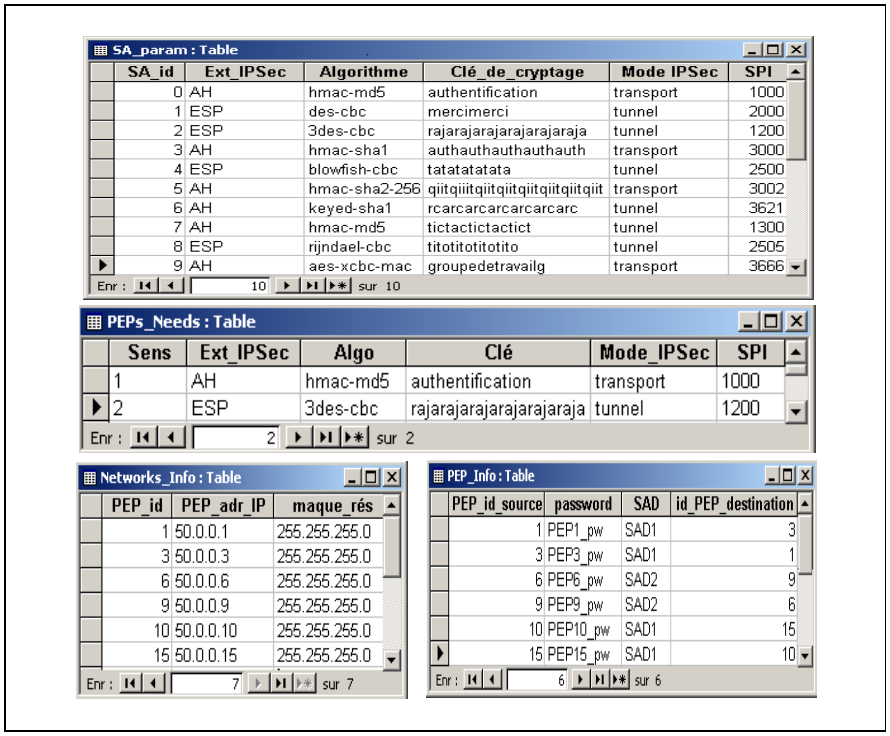


Figure 4. *Adressage IP*

Dans notre implémentation, on a utilisé deux bases de données SADs (*SAD1.mdb* et *SAD2.mdb*), et une base de données *PepAadb.mdb* (PEP Authentication and Authorization Database) (fig.1) qui contient une seule table (PEP_Info) (fig.5). Dans notre exécution, on s'est limité à la base de données *SAD1.mdb* avec ses tables SA_param, Networks_Info et PEP_Needs, et à la base de données *PepAadb.mdb*:



SA_param : Table

SA_id	Ext_IPSec	Algorithme	Clé_de_cryptage	Mode IPSec	SPI
0	AH	hmac-md5	authentification	transport	1000
1	ESP	des-cbc	mercimerci	tunnel	2000
2	ESP	3des-cbc	rajarajarajarajarajaraja	tunnel	1200
3	AH	hmac-sha1	authauthauthauthauth	transport	3000
4	ESP	blowfish-cbc	tatatatatata	tunnel	2500
5	AH	hmac-sha2-256	qiitqiiitqiitqiitqiitqiitqiit	transport	3002
6	AH	keyed-sha1	rcarcarcarcarcarc	tunnel	3621
7	AH	hmac-md5	tictactictactict	tunnel	1300
8	ESP	rijndael-cbc	titotitotitotito	tunnel	2505
9	AH	aes-xcbc-mac	groupedetravailg	transport	3666

Enr : 10 sur 10

PEPs_Needs : Table

Sens	Ext_IPSec	Algo	Clé	Mode_IPSec	SPI
1	AH	hmac-md5	authentification	transport	1000
2	ESP	3des-cbc	rajarajarajarajarajaraja	tunnel	1200

Enr : 2 sur 2

Networks_Info : Table

PEP_id	PEP_adr_IP	maque_rés
1	50.0.0.1	255.255.255.0
3	50.0.0.3	255.255.255.0
6	50.0.0.6	255.255.255.0
9	50.0.0.9	255.255.255.0
10	50.0.0.10	255.255.255.0
15	50.0.0.15	255.255.255.0

Enr : 7 sur 7

PEP_Info : Table

PEP_id_source	password	SAD	id_PEP_destination
1	PEP1_pw	SAD1	3
3	PEP3_pw	SAD1	1
6	PEP6_pw	SAD2	9
9	PEP9_pw	SAD2	6
10	PEP10_pw	SAD1	15
15	PEP15_pw	SAD1	10

Enr : 6 sur 6

Figure 5. *Les tables des bases de données SAD1.mbd et PepAadb.mdb.*

Le contenu de la table SA_param est le résultat de la deuxième invocation de la méthode *modifySADPram()*. Pour ce contenu de la table SA_param, on a invoqué cinq fois successives la méthode *changeSA()* et le contenu de la table PEPs_Needs correspond à la 3ème invocation de cette méthode. L'implémentation, au niveau du PKIServ, de la méthode *modifySADParam()* est:

```
…… // reste du programme
public String  modifySADPram(){
  for (int i = 1; i<=2; i++) { // une boucle pour agir sur les 2 bases de données SAD1.mbd et SAD2.mbd.
    delete(SAD[i]);          // méthode de suppression du contenu de la table SA_param de la base de données considérée.
    changeContenu(SAD[i]); // méthode de renouvellement du contenu de la table SA_param de la base de données sélectionnée.
  }
} …… // reste du programme
```

L'implémentation, au niveau du PKIServ, de la méthode *changeSA()* est:

```
…… // reste du programme
public String changeSA(){
  for (int j =1; j<=2; j++) { // une boucle pour agir sur les 2 bases de données SAD1.mbd et SAD2.mbd
     int r = (int)(Math.random()*10);
     int id_SA= r % 3; // Choix au hasard de l'identifiant de la SA.
     String param = selectSAParam(SAD[j],id_SA); // Sélection des paramètres de la SA considérée.
     Store(SAD[j],param); // méthode de stockage de la SA sélectionnée dans la table PEPs_Needs. }
}…… // reste du programme
```

Pour ces mêmes données de la table PEPs_Needs, le PEP1 a procédé de la manière suivante pour changer sa configuration IPSec : Il demande dans un premier temps au PDP, à travers l'invocation de la méthode *getSAParam()*, la SA à appliquer. Ensuite, le PDP lui authentifie, moyennant la consultation de la table PEP_Info (fig.5) de la base de données *PepAadb.mdb* (fig.1). Enfin, après avoir reçu la SA demandée (fig.5), la nouvelle configuration IPSec de ce PEP est ainsi:

```
Add 50.0.0.1 50.0.0.3 ah 1000 –m transprot –A hmac-md5 ‘‘authentification ’’
Add 50.0.0.3 50.0.0.1 esp 1200 –m tunnel –E 3des-cbc ‘‘rajarajarajarajarajaraja ’’
```

4. Evaluation de notre approche

Le protocole IPSec était l'objectif de beaucoup des travaux de recherches. Ces travaux étaient consacrés soit à l'utilisation du protocole IPSec pour développer des solutions de sécurité soit à l'étude du protocole IPSec lui-même.

Ainsi, Barrère et al. ont présenté une approche (Barrère et al., 2001) de la distribution des politiques de sécurité IPSec et ont également exposé une infrastructure qui repose sur les PDPs, les PEPs, Les MIPs, le protocole COPS-PR, le protocole LDAP et une base de règles de politiques.

Al-Chaal a présenté une approche (Al-Chaal, 2005) dynamique et facilement administrable basée sur la technologie des VPN (IPVPNs : IP Virtual Private Networks). Cette approche est une solution centralisée, où tout, y compris la gestion des services d'administration comme l'adhésion au groupe et la création de topologie VPN, est de la responsabilité d'un centre d'opérations du système (NOS : Network Operation System). Elle contribue également dans la sécurité des services web et dans les techniques de partage de charge et l'amélioration de performances. Enfin,

les mécanismes et les principale utilisations du protocole d'IPSec sont assez discutés et expliqués dans (hsc, 2002)(TCP/IP Guide, 2005).

Notre approche, quant à elle, est à base de politique et repose sur les travaux de la gestion des politiques réalisés au sein d'Imperial College (Imperial College, 2003). Notre solution est caractérisée par l'emploie des domaines, du langage Ponder, d'un système de surveillance, des objets CORBA, d'un PDP et des PEPs. La réunion de ces ingrédients nous a permis de développer une infrastructure dynamique, flexible et extensible pour la gestion de la sécurité des communications inter-domaines. La flexibilité et l'extensibilité de notre approche lui permet d'être applicable dans d'autres architectures et solutions de la sécurité des systèmes distribuées tels que les laboratoires virtuels distribués, les bases de données, les serveurs web, etc.

5. Conclusion

La sécurité est un paramètre important et décisif que l'on doit prendre en considération lors des échanges d'informations entre les domaines et ce à cause de la diversités des attaques (espionnage, destruction et piratage) ainsi que leurs origines. L'assurance de la sécurité des communications inter-domaines exige un effort spécial car les solutions classiques sont maintenant dépassées et la tendance actuelle est l'automatisation de la sécurité moyennant le développement des environnements de gestion très intelligents, flexibles et extensibles.

Notre travail constitue une contribution aux efforts consacrés à ce sujet. Ainsi, on a proposé une infrastructure dynamique de sécurité IPSec permettant de décider et distribuer les SAs IPSec à appliquer dans des délais exacts. De plus, elle permet de changer d'une manière intelligente les contenus des bases de données de sécurité sans aucune intervention de l'homme.

Notre infrastructure de sécurité est caractérisée également par l'emploie du langage Ponder et des domaines ce qui a organisé et facilité plus la gestion de notre environnement de sécurité. En perspective, pour mieux améliorer les performances de notre infrastructure on y introduira les rôles (Lupu, 1993)(Lupu et al., 1997) et on y ajoutera également d'autres protocoles de sécurité.

6. Bibliographie

Al-Chaal L., "Dynamic and Easily Manageable Approach for Secure IP VPN Environments". Ph.D.Dissertation, Institut National Polytechnique de Grenoble, France, 2005.

Barrère F., Benzekri A., Grasset F., Laborde R., Raynaud Y., ''Distribution de politiques de sécurité IPsec'', in : Proc. GRES'01-Gestion de Réseau et de Service,4ème Colloque Francophone, Marrakech-Maroc, pp. 271-284, 2001.

Damianou N., Dulay N., Lupu E., Sloman M., ‘‘The Ponder Policy Specification language’’, in: Proc. Policy 2001, International Workshop on Policies for Distributed Systems and Networks, Bristol, United Kingdom, pp.29-31, 2001.

Damianou N., Dulay N., Lupu E., Sloman M., ‘‘Tools for Domain-Based Management of Distributed Systems’’, in: Proc: IEEE/IFIP Network operations and management symposium (NOMS2002), Florence,Italy, pp.213-218,2002.

El Hamzaoui M., Sekkaki A., Bensassi B., ‘‘The Policy-Based Management of a Virtual Laboratory Security’’, in: Proc. GRES’2005 - Gestion de REseau et de Service, 6ème Colloque Francophone, Luchon-France, pp.183-198, 28 February -03 March 2005a.

El Hamzaoui M., Sekkaki A., Bensassi B., ‘‘Policy-Based Management of the inter-Domain communications Security’’, in: Proc. IEEE/IFIP 4th Latin American Network Operations and Management Symposium (LANOMS), Porto Alegre, Brazil, pp.269-274, 29-31 August, 2005b.

El Hamzaoui M., Sekkaki A., Bensassi B., ‘‘Policy-based Resolution of the Diffie-Hellman protocol vulnerability’’, in: Proc. IEEE/IFIP 4th Latin American Network Operations and Management Symposium (LANOMS), Porto Alegre, Brazil, pp.277-282, 2005c.

Kent S., Atkinson R., ‘‘IP Authentication Header’’, RFC 2402, November 1998a.

Kent S., Atkinson R., ‘‘IP Encapsulating Security Payload (ESP)’’, RFC 2406, November 1998b.

Kent S., Atkinson R., ‘‘Security Architecture for the Internet Protocol’’, RFC 2401, November 1998c.

Lupu E.C., ‘‘A Role-Based Framework for Distributed Systems Management’’, Ph.D. Thesis, Imperial College of Science, Technology and medicine, University of London, UK,1993.

Lupu E., Sloman M., ‘‘ Towards A Role Based Framework For Distributed Systems Management’’, Journal of Network and Systems Management 5, 333-360,1997.

Lymberopoulos L., Lupu E., Sloman M., ‘‘An Adaptive Policy-Based Framework for Network Services Management’’, Journal of Network and Systems Management 11, 277-303, 2003.

Lymberopoulos L., Lupu E., Sloman M., ‘‘PONDER Policy Implementation and Validation in a CIM and Differentiated Services Framework’’, in: Proc. 9th IEEE/IFIP Network Operations and Management Symposium (NOMS 2004), Seoul, Korea, 2004.

Sloman M., Twidle K.P., ‘‘Domains: A Framework for Structuring Management Policy’’, in: Network and distributed systems management’’, Morris Sloman (eds.), Addison-Wesley, pp.433-453, 1994.

Westphall C.B, Sekkaki A., Alvarez L.M., Watanabe W.T., ‘‘Extending TINA Secure On-Line Accounting Services’’, Journal of Network and Systems Management, 11, 2003 .

hsc: http://www.hsc.fr/ressources/articles/ipsec-tech/, last update : 23 October 2002.

TCP/IP Guide: http://www.tcpipguide.com/free/t_IPSecModesTransportandTunnel.htm, last update : September 2005.

Imperial College : http://www.doc.ic.ac.uk/, last update : 2003.