HAL
open science

# Services AAA dans les réseaux adhoc mobiles

Claire Sondès Larafa

## To cite this version:

Claire Sondès Larafa. Services AAA dans les réseaux adhoc mobiles. Autre [cs.OH]. Institut National des Télécommunications, 2011. Français. NNT : 2011TELE0023 . tel-00698490

**HAL Id: tel-00698490**

**https://theses.hal.science/tel-00698490v1**

Submitted on 16 May 2012

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse de doctorat de Télécom SudParis dans le cadre de l'école doctorale S&I
en co-accréditation avec l' Universite d'Evry-Val d'Essonne

Spécialité :
Informatique et réseaux

Par
Mme Claire, Sondès LARAFA

Thèse présentée pour l'obtention du diplôme de Docteur
de Télécom SudParis

# Services AAA dans les réseaux ad hoc mobiles

Soutenue le 21 octobre 2011 devant le jury composé de :

| | | |
|---|---|---|
| Mme Houda Labiod | Maître de conférence, HDR<br>TELECOM ParisTech | rapporteur |
| M. Christophe Bidan | Maître de conférence, HDR<br>Supélec | rapporteur |
| M. Michel Barbeau | Professeur<br>Carleton University | examinateur |
| M. Didier Begay | Risk Manager des offres, Dr.<br>Orange | examinateur |
| M. Abdelmalek Benzekri | Professeur<br>IRIT, Université Paul Sabatier | examinateur |
| Mme Maryline Laurent | Professeur<br>TELECOM SudParis | directrice de thèse |

Thèse n° 2011TELE0023

# Remerciements

Je ne voudrais pas que le caractère nécessairement convenu de cette page de remerciements pût masquer l'ampleur de ma reconnaissance non plus que la force de ma gratitude.

Cette gratitude s'adresse d'abord au professeur Maryline Laurent pour avoir accepté de diriger cette thèse : elle m'en a fourni le sujet, a garanti les conditions matérielles, scientifiques et morales de son déroulement, m'a accordé une autonomie et une confiance totale, tout en sachant me réconforter avec discrétion dans les moments de doute.

Mes vifs remerciements vont à M. Richard Nock, professeur à l'Université Antilles-Guyane, laboratoire CEREGMIA, pour avoir accepté, alors que j'habitais encore la Martinique, d'assurer la co-direction de mes recherches à leur début.

Mme Houda Labiod, maître de conférence HDR à Télécom ParisTech et M. Christophe Bidan, maître de conférence HDR à Supélec ont tous deux accepté la tâche de rapporter sur mes travaux, qu'ils soient ici assurés de ma reconnaissance.

Je remercie M. Michel Barbeau, professeur à Carleton University et M. Abdelmalek Benzekri, professeur à l'université Paul Sabatier de l'honneur qu'ils me font en participant à ce jury, comme je remercie M. Didier Begay, Risk Manager des offres chez Orange, d'avoir accepté d'y apporter le regard d'un industriel.

Un travail de recherche est indissociable des laboratoires où il a pris jour. A ce titre, je remercie le laboratoire CEREGMIA où mes travaux ont débuté ainsi que son directeur le professeur Fred Célimène pour la chaleur de son accueil.

C'est ensuite à Télécom SudParis (SAMOVAR, LOR, RST) que j'ai eu la possibilité de poursuivre mes travaux. Je tiens à en remercier les responsables pour la qualité de l'ambiance qu'ils savent y entretenir. Je remercie le professeur Hakima Chaouchi pour ses nombreux conseils ; Mme Brigitte Laurent, secrétaire du département LOR, pour son aide efficace ; mes collègues : Jérôme, Providence, Kheira, Andrei, Ethmane, Léon et les autres qui ont entretenu une ambiance de recherche et avec qui j'ai eu de longues discussions autour de repas et à l'occasion de formations.

Je tiens à remercier vivement tous ceux qui maintiennent en ligne des ressources de qualité comme le TLFi, aide considérable à l'utilisation de la langue française, et les « Petites leçons de typographie » qui m'ont aidée dans mes tentatives de rendre cette thèse agréable à lire. Merci aussi à l'équipe Toilers de l'école des Mines du Colorado pour les ressources dont ils font profiter la communauté.

La vie d'un doctorant propose de multiples circonstances où il enseigne, encadre des stagiaires. Ce fut pour moi l'occasion de travailler auprès d'étudiants. Sans pouvoir en dresser la liste, je voudrais remercier chacun pour les riches échanges noués lors de ces rencontres.

J'exprime mes remerciements les plus chaleureux à tous mes amis et mes proches pour leurs encouragements et pour m'avoir supporté toutes ces années, enfin à Alain s'il ne sait pas pourquoi moi je le sais.

# Abstract

Mobility is an important component of people's liberty. The evolution of technological means contributes to its enhancement. In addition to the transport issue, the question of keeping people connected is, in that context, particularly significant. Technological change strained the notion of telecommunications networks in the sense that scattered or clustered but mobile individuals had to remain in touch with others electronically. From the age of analogue networks to the digital networks era, from wired networks to wireless and mobile networks, technology has never stopped evolving. The last decades have witnessed the emergence of digital and wireless networks where not only the users, but also the network infrastructure to which they contribute, are mobile. These networks are spontaneously formed and autonomously maintained. They are termed *Mobile Ad hoc Networks* (MANETs), in contradistinction to infrastructure networks.

Security is a general concern of human beings. They feel the need for it when using a network, too. This need is particularly glaring when it comes to exchanging critical, financial or strategic data. Confidentiality of communications, source authentication, integrity assurance, prevention of repudiation, are all objectives to be achieved. Various security solutions have been devised in this context as wired networks were then adapted to wireless and mobile networks. AAA (*Authentication, Authorization and Accounting*) frameworks are among these solutions. They are generally used for commercial purposes, which raises financial issues – and we all know how much important they are.

Due to their ease of deployment and inexpensive implementation, MANETs, first used in the military field, seem to have a future in commercial applications. That is why the present thesis proposes to design an AAA service that is adapted to the characteristics of such networks.

In this perspective, the thesis examines, to begin with, mobile ad hoc networks in order to understand their characteristics and potentials. It then probes the security solutions that exist in infrastructure networks, with special emphasis on those dealing with access control and AAA services. The AAA solutions for MANETs proposed up to now are subsequently analyzed and classified in order to determine their strengths and weaknesses.

This in-depth study led to the design, in the second part of the thesis, of an AAA service that meets the expectations identified. It is a distributed service intended to answer the needs of autonomous operations in MANETs when a nearby operator is absent. It features several detailed authentication and authorization mechanisms and protocols with an authentication protocol simultaneously involving one or more parties. These protocols are designed such that they can be easily integrated to the IPv6 protocols suite. Moreover, their security is discussed – in particular, that of the authentication protocol thanks to a

formal validation tool.

Unlike the communication mode of the authorization protocols, that of the authentication protocol is one-to-many, which led us to study more deeply its performances thanks to modeling using explicit mathematical computations and to simulations techniques. The obtained results showed that the protocol scales for a MANET including a hundred nodes. Under certain conditions that we explained, its performances, in terms of the probability of authentication success and the length of the executing time, reach optimal values.

# Résumé

La mobilité est une composante importante de la liberté des personnes. L'évolution des moyens technologiques y contribue au premier chef. Outre la question du transport, celle du maintien du lien entre les individus est en ce sens particulièrement prégnante. Elle a mis à rude épreuve la notion de réseaux de télécommunications puisqu'il s'agit de répondre, pour des individus éparpillés ou concentrés, mais mobiles, au besoin de rester reliés. De l'ère des réseaux analogiques à celle des réseaux numériques, de l'ère des réseaux filaires à celle des réseaux sans-fil et mobiles, la technologie n'a cessé d'évoluer. Ces dernières décennies ont vu apparaître des réseaux numériques sans-fil, où non seulement il y a mobilité des utilisateurs mais aussi mobilité de l'infrastructure du réseau à laquelle ils contribuent. Ces réseaux se constituent de façon spontanée. Ils se maintiennent de manière autonome. On les désigne par le terme *réseaux ad hoc mobiles* (en anglais *Mobile Ad hoc Networks* ou MANET) qui s'oppose naturellement à celui de *réseaux à infrastructure*.

La sécurité est une préoccupation générale des êtres humains. Ils en ressentent aussi le besoin en matière de réseaux. Ce besoin est particulièrement criant lorsque sont échangées des données critiques, financières ou stratégiques. La confidentialité des échanges, l'authentification des sources, l'assurance d'intégrité, la prévention de la récusation sont autant d'objectifs qu'il faut alors atteindre. Diverses solutions de sécurité ont été conçues dans cette optique pour les réseaux filaires puis ont ensuite été adaptées aux réseaux sans-fil et mobiles. Les architectures AAA (*Authentication, Authorization, Accounting*) en font partie. Elles sont en général utilisées dans un contexte commercial.

Tant par leur facilité de déploiement que par la baisse des coûts de mise en œuvre qu'ils engendrent, les réseaux ad hoc mobiles, après avoir bien servi dans le domaine militaire, semblent avoir un avenir dans les applications commerciales. C'est pourquoi, nous nous proposons dans cette thèse de concevoir une architecture AAA adaptée aux spécificités de ces réseaux.

Nous étudions d'abord les réseaux ad hoc mobiles et leurs caractéristiques. Ensuite, nous présentons les solutions de sécurité qui existent dans les réseaux à infrastructure. Nous examinons, en particulier, les solutions qui permettent le contrôle d'accès et dont sont engendrées les architectures AAA. Les solutions AAA proposées pour les MANETs sont par la suite analysées et classifiées afin de déterminer les manques et les vulnérabilités.

Cette étude approfondie nous amène à proposer une architecture AAA répondant aux attentes identifiées. C'est une architecture distribuée qui répond, en particulier, au besoin d'autonomie des opérations dans les MANETs et où les protocoles exécutés peuvent impliquer simultanément plus de deux parties. Un ensemble de protocoles et de mécanismes d'authentification et d'autorisation

s'intégrant avec la suite des protocoles IPv6 a été proposé. Leur sécurité a été discutée. Celle, en particulier du protocole d'authentification a fait l'objet de validation formelle.

Contrairement aux protocoles utilisés dans la phase d'autorisation des services AAA proposés, le mode de communication *multi-parties* et *multi-sauts* du protocole d'authentification nous a poussé à mener une analyse approfondie de ses performances. Pour cela, nous avons eu recours, dans un premier temps, à la modélisation au moyen de calculs mathématiques explicites ensuite à la simulation. Les résultats obtenus montrent que ce protocole passe à l'échelle d'un MANET comprenant au moins cent nœuds. Dans certaines conditions d'implémentation que nous avons définies, ses performances, tant celle liée à sa probabilité de terminaison avec une issue favorable [1] que celle portant sur son temps d'exécution, atteignent des valeurs optimales.

1. La probabilité de terminaison du protocole d'authentification est la probabilité de réussite d'une authentification

# Table des matières

**Introduction** **xix**
Réseaux et mobilité . . . xix
Mobilité et sécurité, des objectifs incompatibles ? . . . xxii
Organisation du mémoire . . . xxiii

**I Réseaux ad hoc mobiles et exigences des protocoles AAA : un tour de l'existant** **1**

**1 Réseaux ad hoc et problèmes de sécurité** **3**
1.1 Introduction . . . 3
1.2 Aperçu . . . 4
1.2.1 Définition et vocabulaire . . . 4
1.2.2 Paradigme . . . 5
1.2.3 Le routage : un problème fondamental . . . 7
1.3 Applications : entre passé et futur . . . 9
1.3.1 Le départ : les militaires . . . 9
1.3.2 Évolution : les civiles . . . 12
1.3.3 Civiles non commerciales . . . 14
1.3.4 Civiles commerciales . . . 15
1.4 Défis . . . 17
1.4.1 Liens sans-fil . . . 17
1.4.2 Réseau multi-sauts et nœuds mobiles . . . 18
1.4.3 Absence d'une administration centrale . . . 18
1.4.4 Contexte distribué . . . 19
1.4.5 Nœuds égoïstes . . . 19
1.4.6 Nœuds malveillants . . . 19
1.4.7 Contrainte d'énergie . . . 20
1.4.8 Terminaux hétérogènes . . . 20
1.5 Conclusion . . . 20

**2 Sécurité des réseaux à infrastructure** **21**
2.1 Introduction . . . 22
2.2 Vocabulaire et méthodologie . . . 22
2.2.1 Définition de la sécurité . . . 22
2.2.2 Méthodologie . . . 22
2.2.3 Que veut-on sécuriser ? . . . 24
2.2.4 Contre quels dangers ? . . . 26

2.2.5 Comment ? . . . . . 27
2.3 Techniques de contrôle d'accès . . . . . 28
2.3.1 Histoire . . . . . 28
2.3.2 Droits d'accès et domaine . . . . . 30
2.3.3 Identification contre Authentification . . . . . 30
2.3.4 Authentification par mot de passe statique . . . . . 31
2.3.5 Authentification par mot de passe dynamique (OTP) . . . 32
2.3.6 Garde-barrières . . . . . 32
2.4 Outils cryptographiques et services de sécurité . . . . . 34
2.4.1 Algorithmes de chiffrement . . . . . 35
2.4.2 Hachage cryptographique . . . . . 38
2.4.3 Services de sécurité . . . . . 39
2.5 Relation de confiance et gestion de clés . . . . . 40
2.5.1 Hiérarchisation de la confiance . . . . . 41
2.5.2 Renforcement de la vigilance en environnement distribué . 42
2.5.3 Partage de clés . . . . . 43
2.6 Protocoles de sécurité . . . . . 44
2.7 Conclusion . . . . . 45

**3 Taxonomie des services AAA dans les MANETs 47**
3.1 Introduction . . . . . 48
3.2 Vue générale de AAA . . . . . 48
3.2.1 Modèle d'architecture . . . . . 48
3.2.2 Protocoles . . . . . 50
3.2.3 Processus et mécanismes . . . . . 51
3.3 Taxonomie . . . . . 54
3.3.1 Modèle de dépendance . . . . . 55
3.3.2 Modèle d'architecture . . . . . 57
3.3.3 Services commercialisables et comptabilité . . . . . 61
3.3.4 Modes de communication AAA . . . . . 64
3.3.5 Synthèse . . . . . 68
3.4 Considérations annexes . . . . . 70
3.4.1 Bases de données . . . . . 70
3.4.2 Architecture matérielle et logicielle . . . . . 71
3.5 Discussion . . . . . 71
3.6 Conclusion . . . . . 73

**II Conception de services AAA distribués 75**

**4 Contribution : architecture, protocoles et opérations AAA distribués 77**
4.1 Introduction . . . . . 77
4.2 Cahier des charges . . . . . 78
4.2.1 Architecture . . . . . 78
4.2.2 Besoins et choix protocolaires . . . . . 78
4.2.3 Outils cryptographiques . . . . . 79
4.3 Architecture proposée . . . . . 79
4.3.1 Description et vocabulaire . . . . . 79
4.3.2 Initialisation . . . . . 81

4.3.3 Évolution . . . . . 83
4.4 Protocole d'authentification proposé . . . . . 84
4.4.1 Authentification mutuelle avec ISO 9798-3 . . . . . 84
4.4.2 Distribution . . . . . 85
4.4.3 Proposition . . . . . 88
4.4.4 Optimisation du nombre d'étapes . . . . . 94
4.4.5 Algorithme à multiples tentatives d'authentification . . . 95
4.5 Protocoles d'autorisation proposés . . . . . 97
4.5.1 Jeton d'accès . . . . . 97
4.5.2 Autorisation au service de routage . . . . . 98
4.5.3 Autorisation au service de contenus . . . . . 101
4.5.4 Collectes d'informations pour les besoins de comptabilité 103
4.6 Conclusion . . . . . 103

**5 Validation formelle de la sécurité du protocole AAA d'authentification 105**
5.1 Introduction . . . . . 105
5.2 Vérification formelle avec AVISPA . . . . . 106
5.2.1 Modélisation en HLPSL . . . . . 106
5.2.2 Modèle de Dolev-Yao . . . . . 107
5.2.3 L'outil SPAN . . . . . 107
5.2.4 Preuves de sécurité . . . . . 107
5.3 Spécification du protocole d'authentification . . . . . 108
5.3.1 Protocole ISO 9798-3 . . . . . 108
5.3.2 Notre protocole d'authentification . . . . . 108
5.4 Exploitation des résultats . . . . . 112
5.4.1 Protocole ISO 9798-3 . . . . . 112
5.4.2 Notre protocole . . . . . 114
5.5 Conclusion . . . . . 118

**III Évaluation du protocole 121**

**6 Évaluation du temps d'authentification 123**
6.1 Introduction . . . . . 123
6.2 Modélisation et simulation : terminologie . . . . . 124
6.3 Quelques définitions . . . . . 125
6.4 Modèle mathématique et calculs . . . . . 127
6.4.1 Calcul du temps de transfert sur un lien à un seul saut . . 128
6.4.2 Calcul du temps de transfert sur un lien à sauts multiples 130
6.4.3 Prise en compte du hasard . . . . . 130
6.4.4 Calcul du temps d'authentification . . . . . 133
6.4.5 Résultats . . . . . 135
6.5 Modèles emboîtés et simulations . . . . . 138
6.5.1 Considérations techniques, paramétrages et conditions de validation . . . . . 138
6.5.2 Scénario statique . . . . . 139
6.5.3 Ajout de mouvements déterministes . . . . . 143
6.5.4 Prise en compte des temps de calculs cryptographiques . . 147
6.6 Conclusion . . . . . 147

**7 Réglage fin et performances du protocole 149**
7.1 Introduction . . . 150
7.2 Étude théorique : optimisation du taux de réussite par ajustement de seuils . . . 151
7.2.1 Choix du seuil cryptographique $th$ . . . 151
7.2.2 Introduction de seuils intermédiaires . . . 152
7.2.3 Choix de seuils intermédiaires optimaux . . . 152
7.3 Choix du simulateur et des modèles . . . 155
7.3.1 Simulation de réseaux : aperçu . . . 155
7.3.2 Choix du simulateur . . . 156
7.3.3 Modèles utiles et classification . . . 162
7.4 Travail préparatoire aux simulations dans NS-2 . . . 166
7.4.1 Organisation des outils et des données . . . 166
7.4.2 Extension de NS-2 par une implémentation du protocole d'authentification . . . 168
7.5 Scénarios de simulations et résultats . . . 173
7.5.1 Paramétrages généraux . . . 174
7.5.2 Paramétrage du trafic AAA d'authentification . . . 176
7.5.3 Scénarios . . . 178
7.5.4 Résultats . . . 179
7.6 Réglages de l'algorithme à multiples tentatives d'authentification 184
7.6.1 Estimation du temps d'attente maximum $t_{max}$ . . . 184
7.6.2 Réglage du nombre maximum de tentatives $K$ . . . 185
7.7 L'entrelacement des phases est-il une amélioration ? . . . 185
7.7.1 Contribution de l'approche d'*entrelacement* par rapport à l'approche de *séparation* . . . 186
7.8 Conclusion . . . 187

**Conclusion et perspectives 188**

**A AODV 193**

**B Rappels de cryptographie 197**
B.1 Polynômes . . . 197
B.1.1 Convention pour les produits de plusieurs facteurs . . . 197
B.1.2 Interpolation de Lagrange . . . 198
B.2 Groupes cycliques . . . 199
B.2.1 Exponentiation dichotomique . . . 200
B.2.2 Exemples utiles . . . 200
B.3 Tronçons et Codons . . . 201
B.4 Problèmes difficiles et fonctions à sens unique . . . 202
B.5 Factorisation et système RSA . . . 203
B.5.1 Envois chiffrés et discrétion . . . 203
B.5.2 Envois signés et authentification . . . 203
B.6 Logarithme discret et système El-Gammal . . . 204
B.7 D'autres systèmes . . . 205

**C SEcure Neighbor Discovery (SEND) 207**
C.1 Internet Protocol version 6 (IPv6) . . . . . 207
C.2 Neighbor Discovery Protocol (NDP) . . . . . 209
C.3 SEND . . . . . 210
C.3.1 Cryptographically Generated Address (CGA) . . . . . 211
C.3.2 Options SEND . . . . . 212

**D Mécanisme d'accès de base de la fonction DCF 215**

**E Calculs avec Maple 219**
E.1 Modèle max . . . . . 219
E.2 Modèle somme . . . . . 260

**Bibliographie 265**

**Index 275**

# Table des figures


1 Évolution du nombre d'abonnés mobiles dans la zone de l'OCDE [2]. xx

1.1 Protocoles de routages pour les MANET définis au sein de l'IETF 9
1.2 L'Internet tactique. *Source : Thalès* . . . . . 11
1.3 Évolution des soumissions à l'IEEE . . . . . 13
1.4 Réseaux 4G [1] . . . . . 16
1.5 Phénomène de la station cachée . . . . . 18

2.1 Modèle en oignon . . . . . 29
2.2 Authentification cryptographique . . . . . 39
2.3 Procédés de partage de clés cryptographiques . . . . . 43
2.4 Divers types de communications chiffrées (Les cadenas indiquent les communications chiffrées) . . . . . 44

3.1 Architecture AAA simple . . . . . 49
3.2 Modèle des architectures AAA . . . . . 49
3.3 Enchaînement des mécanismes AAA . . . . . 51
3.4 Architecture de protocoles proposée . . . . . 53
3.5 MANET tributaire . . . . . 56
3.6 Diagramme d'états d'un nœud ad hoc . . . . . 58
3.7 MANET totalement tributaire à architecture AAA hiérarchique centrale : diagramme de séquences . . . . . 65
3.8 MANET partiellement tributaire à architecture AAA hiérarchique centrale : diagramme de séquences . . . . . 67
3.9 MANET partiellement tributaire à architecture AAA pair-à-pair : diagramme de séquences . . . . . 70

4.1 *Dist-AAA* : Architecture AAA destinée à fonctionner dans un MANET . . . . . 80
4.2 Authentification mutuelle du standard ISO 9798-3 . . . . . 84
4.3 Hergé, *Le Secret de La Licorne*, (Les Aventures de Tintin). Superposition des trois parchemins et résolution de l'énigme : les coordonnées de l'île où se cache le trésor du pirate Rackham le Rouge sont ainsi découverts . . . . . 87
4.4 Protocole d'authentification distribué . . . . . 90
4.5 Optimisation du protocole proposé . . . . . 95
4.6 Algorithme à multiples tentatives d'authentification . . . . . 96
4.7 Algorithme de vérification de la validité d'un jeton d'accès . . . . 100

4.8 Attaque par vol du jeton . . . . . 101
4.9 Format d'un message SEND étendu (Ext-SEND) . . . . . 102

5.1 Animation SPAN . . . . . 108
5.2 Modélisation de l'envoi de $ID_{JN}$ (jn) et de $R_{AAA}$ (*nonce*) . . . . 109
5.3 Modélisation du protocole pour $th = n = 3$ . . . . . 111
5.4 Résultat donné par AVISPA . . . . . 112
5.5 Détection de l'attaque . . . . . 113
5.6 Attaque menée par l'intrus . . . . . 114
5.7 Résultat donné par AVISPA . . . . . 115
5.8 Animation SPAN . . . . . 116
5.9 Résultat donné par AVISPA . . . . . 117
5.10 Résultat donné par AVISPA . . . . . 118

6.1 Protocole d'authentification à 4 types de messages . . . . . 126
6.2 Route à 3 sauts, de longueur 3 ; nœuds également espacés . . . . 127
6.3 Graphe étoilé à branches d'égales longueurs : JN est à la même longueur de route de chaque pair AAA . . . . . 128
6.4 Étapes de transmission entre JN et $srv_j$ distants d'un seul saut et temps associés . . . . . 129
6.5 Route à 3 sauts, de longueur 3 ; nœuds également espacés . . . . 130
6.6 Temps moyen d'authentification dans le modèle *max* . . . . . 137
6.7 Placement des nœuds dans le plan de simulation ($n = 3$, $hops = 3$) 139
6.8 Temps moyen d'authentification dans le scénario statique . . . . 140
6.9 Temps d'authentification moyen dans le modèle *somme* . . . . . 142
6.10 Trajectoire des nœuds relais . . . . . 144
6.11 Influence du mouvement des nœuds relais sur la connectivité du graphe du réseau. A gauche : la distance minimale entre le nœud 21 et les nœuds 11, 12 et 13. A droite : la distance minimale entre le nœud 31 et les nœuds 21 et 22 . . . . . 145

7.1 A gauche : $\widehat{P_2}$ lorsque $th = 5$, $th_M \in \{5, ..., 20\}$. A droite : $\widehat{P_2}$ lorsque $th = 10$, $th_M \in \{10, ..., 20\}$ . . . . . 154
7.2 Diagramme de flux . . . . . 167
7.3 Nouvelles sous-hiérarchies dans la hiérarchie de classe de NS-2 . . 169
7.4 Diagramme d'état du numéro de séquence interne de *AAAClient* 172
7.5 Positions initiales des clients et des pairs . . . . . 175
7.6 Détermination de la saturation du réseau : TSR2D vs. $r$ . . . . . 176
7.7 Nombre de messages d'authentification envoyés cumulés . . . . . 179
7.8 Authentifications réussies cumulées . . . . . 180
7.9 Nombre de messages d'authentification perdus cumulés . . . . . 181
7.10 Ratios d'authentifications échouées à la première phase et d'authentifications échouées à la deuxième phase . . . . . 181
7.11 Nombres moyens de pairs ayant répondu pendant une authentification échouée dès la première phase et pendant une authentification échouée à la deuxième phase . . . . . 182
7.12 Taux de réussite . . . . . 183
7.13 Cumulated AAs for $th = 5$ and $th = 10$. On the left-hand side : *separation case*. On the right-hand side : *interleaving case* . . . . 184

7.14 $th_M = 4$, $th_i = 3$ and $th = 2$. A gauche : cas de *séparation*. A droite : cas *entrelacement* . . . . . 186

A.1 Envoi de RREQ par la source $S$ et réception de deux réponses $\text{RREP}_4$ du nœud 4 et $\text{RREP}_5$ du nœud 5 . . . . . 194

B.1 Tracés de $\mathcal{L}_5$ aux points $(1, y), (1.22, 2), (4, -10), (6.7, 2), (7.1, 6)$. 198
B.2 Un groupe cyclique s'identifie à l'orbite de $g$ . . . . . 199
B.3 Loi de composition $R = P \cdot Q$ sur deux courbes elliptiques : A gauche $W$ a trois racines, à droite une seule . . . . . 201

C.1 Adresse IPv6 . . . . . 208
C.2 Format d'un message NDP . . . . . 209
C.3 Dialogue NDP entre nœuds voisins . . . . . 210

D.1 Exemple illustrant le mécanisme d'accès de base . . . . . 216

# Liste des tableaux


1.1 Diverses déclinaisons de réseaux ad hoc . . . . . 6
1.2 Applications civiles . . . . . 13

3.1 Taxonomie des solutions AAA dans les MANETs . . . . . 69

4.1 Résumé des notations dans *Dist-AAA* . . . . . 81
4.2 Résumé des notations . . . . . 89

6.1 Calculs de probabilités compte tenu du schéma de temporisation exponentiel . . . . . 132
6.2 **Résumé des hypothèses** . . . . . 135
6.3 Valeurs des paramètres utilisés dans le modèle . . . . . 136
6.4 Commandes NS et paramètres de simulations . . . . . 140

7.1 Valeurs des paramètres . . . . . 153
7.2 Récapitulatifs des valeurs optimales choisies et du taux de réussite correspondant . . . . . 154
7.3 Tableau comparatif des simulateurs de réseaux les plus connus . 157
7.4 Choix de modèles, protocoles et paramètres associés . . . . . 166
7.5 Protocoles et paramètres associés (suite) . . . . . 171
7.6 Résumé des paramétrages . . . . . 174
7.7 Choix pour les scénarios de simulation . . . . . 178
7.8 Fraction d'authentifications supplémentaires abouties en fonction du nombre de tentatives supplémentaires . . . . . 185
7.9 Comparison of the *interleaving* vs *separation* approaches . . . . . 186

D.1 Valeurs des fenêtres de contention pour les trois types de couche physique spécifiées par le standard IEEE 802.11 : FHSS (Frequency-Hopping Spread Spectrum), DSSS (Direct Sequence Spread Spectrum), et IR (Infra-Red) . . . . . 216

# Introduction

## Réseaux et mobilité

Dans notre société la mobilité semble faire partie intégrante de la liberté des personnes. Depuis Toumaï et Lucy, l'humanité n'a cessé de se disséminer sur les cinq continents. Les moyens de transport de matières, inventés durant la révolution industrielle dans les pays occidentaux, ont été vite transformés en moyen de transport de passagers. Aujourd'hui, sans aller jusqu'au thème Chatwinien de l'« l'horreur du domicile [3] », la mobilité fait partie de notre vie quotidienne, quand nous partons au travail ou que nous nous déplaçons pour des besoins quotidiens ou encore quand nous voyageons dans des pays étrangers. Le monde, depuis le IX$^{\text{ème}}$ siècle, est devenu, selon l'expression de Marshall McLuhan, un village planétaire (*Global Village*) [4].

Pourtant, les personnes gardent le besoin d'être reliées entre elles malgré leur mobilité, et cela afin de rester informées, de travailler en groupe, de garder des liens sociaux, etc. Le mode d'organisation en réseaux semble, en effet, s'imposer dans tous les compartiments de notre société : les réseaux d'entreprises, les réseaux sociaux, les réseaux associatifs ou de lobbying, etc. L'existence de moyens matériels adéquats issus de la modernité et parmi eux, ceux qui, comme les technologies sans-fil confèrent liberté et mobilité, consolide et encourage même cette organisation en réseaux.

Dans la société nomade où nous vivons, l'utilisateur supporte de plus en plus mal de ne pas avoir à tout instant et en tout lieu accès aux services dont il dispose chez lui ou sur son lieu de travail. Cette tendance se trouve encouragée par la multiplication et la diversification des appareils électroniques et des services fournis dans ce cadre.

Nous assistons, en effet, de nos jours, à une prolifération sans précédent des appareils électroniques munis d'une liaison sans-fil. Cette évolution quantitative amène à un changement qualitatif majeur, une révolution dans notre société de l'information. Les ordinateurs portables, les téléphones portables, les assistants personnels, les imprimantes, les radios et lecteurs de CD, les enceintes sonores, les disques durs, les lecteurs de musique, les localisateurs GPS, donnent de nombreux exemples d'appareils dotés de liaisons sans-fil que l'on trouve aujourd'hui sur le marché des produits informatiques. Ces appareils sont non seulement de plus en plus petits et de plus en plus puissants, mais aussi de moins en moins chers.

3. Bruce Chatwin, *Anatomy Restlessness*, isbn 0-224-04292-0.

4. *War and Peace in the Global Village*, 1967, traduction française : *Guerre et Paix dans le village planétaire*, Robert Laffont, Paris, 1970.

D'autre part, la téléphonie mobile était parmi les tout premiers des services mobiles et certains se rappellent ou peuvent redécouvrir ces publicités des années 90 où une personne au statut social manifestement enviable appelait un parent ou un ami d'une terrasse de café ou d'une station de vacances et commençait par la phrase rituelle « devine d'où je t'appelle ».

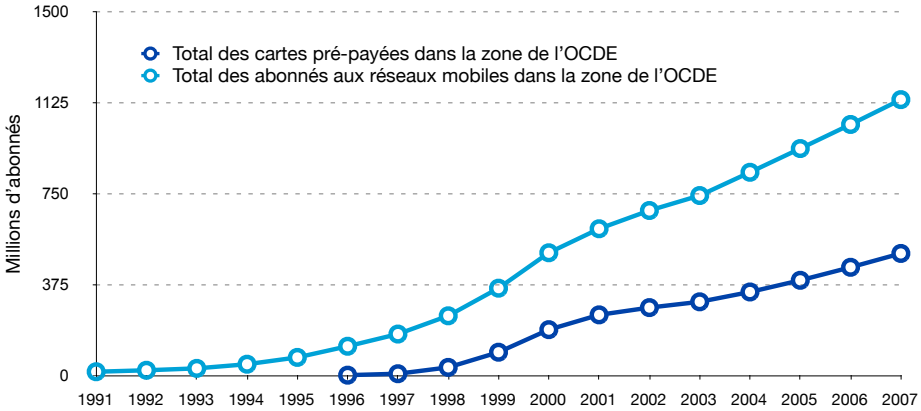


FIGURE 1 – Évolution du nombre d'abonnés mobiles dans la zone de l'OCDE[4].

Aujourd'hui, d'après une étude effectuée par l'OCDE [2], elle est en train de prendre le pas sur la téléphonie fixe. En effet, selon cette étude, le nombre d'abonnés aux services mobiles ne cesse de croître et cela de façon importante (cf. FIGURE 1). Le nombre d'abonnements aux services mobiles a augmenté dans la zone OCDE au taux annuel de 10% au cours des deux années 2006 et 2007, pour atteindre 1.13 milliard en 2007, ce qui équivaut à un taux de pénétration effectif de 96.1 abonnés mobiles pour 100 habitants. De plus, le nombre de minutes de communication augmente sur les réseaux mobiles tandis qu'il diminue sur les réseaux fixes. Les données relatives à la période 2005-2007 [2] tendent à montrer que le trafic de communications nationales sur réseau fixe baisse dans la plupart des pays de la zone.

D'autres services accessibles à des utilisateurs mobiles sont apparus plus tard, comme la navigation sur internet, le téléchargement du courrier, le transfert de fichiers, la synchronisation de données, la situation d'une position géographique sur une carte, la diffusion de flux vidéo, etc. Aujourd'hui, la majeure partie de ces services est accessible à travers des réseaux basés sur une infrastructure fixe comme celle des réseaux cellulaires, des réseaux locaux sans-fils et des réseaux locaux filaires.

Bien que les réseaux basés sur une infrastructure fixe présentent un excellent moyen pour assurer les services en mobilité, leur mise en place peut néanmoins s'avérer longue, complexe, et potentiellement coûteuse. Dans certaines conditions géographiques ou quand il y a un besoin temporaire et particulier voire urgent de réseau, assurer une connectivité réseau peut ainsi constituer un défi à la fois technologique et financier.

4. Source : OCDE, Technologie de l'information et des communications, Perspectives des communications de l'OCDE 2009.

Traditionnellement, on distingue dans un réseau les équipements destinés aux utilisateurs, et nous en avons évoqué des exemples ci-dessus, de ceux destinés à assurer la communication : centraux téléphoniques, stations cellulaires, relais de transmission, satellites artificiels de communication. Ces équipements coûtent en général cher et sont gérés par un ou plusieurs opérateurs plus ou moins spécialisés. La multiplication des supports, en amenant la multiplication de ces infrastructures augmente de façon considérable les coûts à la charge des exploitants des réseaux.

Il était donc naturel pour ces opérateurs de rechercher à faire assurer tout ou partie de la tâche d'acheminement du trafic par les dispositifs achetés à d'autres fins par les utilisateurs, allégeant d'autant leurs coûts d'infrastructure. A la limite extrême, un réseau pourrait ainsi se constituer sans aucune infrastructure par les seules capacités des appareils des utilisateurs de ce réseau, un peu comme pourrait l'être un réseau de talkies-walkies, sans aucune station de base. Un tel réseau s'appelle un réseau *ad hoc*. Nous consacrerons une importante partie de cette thèse à ce type de réseau et en particulier à la catégorie mobile c'est-à-dire les *réseaux ad hoc mobile* souvent abrégés en MANET pour *Mobile Ad hoc NETworks*.

La combinaison de l'absence d'équipement d'infrastructure et celle de liaisons sans fil permet de déployer sans difficulté des réseaux ad hoc. Elle permet aussi d'envisager simplement de modifier rapidement la disposition des appareils des utilisateurs sur le terrain, l'ensemble se reconfigurant de façon transparente et automatique. Grâce à ce type de réseau, les terminaux peuvent non seulement communiquer directement entre eux, mais aussi bénéficier de connexions aux services mobiles à travers un terminal spécifique qui opérerait comme point d'accès à un réseau basé sur une infrastructure. Les services mobiles se trouvent ainsi étendus à la partie sans infrastructure c'est-à-dire au réseau ad hoc. Ainsi, cela permet d'envisager de répondre simplement au besoin croissant de services en mobilité des utilisateurs à moindre frais.

Ces réseaux continuent d'évoluer au fur et à mesure qu'apparaissent de nouvelles technologies sans-fil. Ils tendront à devenir plus flexibles et plus puissants avec des capacités ad hoc plus intéressantes. Les chercheurs académiques, les organismes de standardisation et les industriels, qui leur portent un intérêt croissant, font beaucoup d'efforts en ce sens.

Dans cette perspective, plusieurs services mettant à profit les réseaux ad hoc ont vu le jour, d'abord dans le domaine militaire et de secours en cas d'urgence, ensuite dans le domaine civile à travers des produits commerciaux. En effet, des applications grand public avaient déjà commencé à être demandées, par exemple les jeux en réseau, le téléchargement vidéo, les services d'informations aux voyageurs, l'assistance sur la route, l'organisation de classes ou bureaux virtuels, etc. Même les appels téléphoniques qui passent par les réseaux fixes et les réseaux cellulaires pourront être acheminés sur une partie de leur trajet par des réseaux ad hoc car d'ores et déjà, les réseaux de convergence qui rapprochent le téléphone, les réseaux de données et les flux vidéo, dits *réseaux 4G*, incluront ces réseaux.

## Mobilité et sécurité, des objectifs incompatibles ?

La sécurité est un besoin de l'être humain, cité d'ailleurs dans l'article 22 de la déclaration des droits de l'homme [5]. La pyramide des besoins de Maslow cite ce besoin comme important [6].

Comme tous les êtres humains, les usagers d'un réseau ressentent ce besoin de sécurité et cela de façon d'autant plus importante que l'utilisation du réseau a une plus grande place dans leur vie. Les données transmises se rapportant à des biens financiers et intellectuels nécessitent la confidentialité des échanges, l'authentification de leurs sources, la sûreté de leur acheminement, l'assurance de leur intégrité, la prévention du déni d'agissements antérieurs, etc. Ces apports ne peuvent, par ailleurs, être fournis sans la garantie de la disponibilité du réseau et des services associés. On ne peut en effet concevoir la sécurité sans la sûreté de fonctionnement.

Des solutions de sécurité ont été mises en œuvre, d'abord dans les réseaux filaires, ensuite dans les réseaux sans-fil au fur et à mesure que ces derniers quittaient le stade expérimental. Elles reposent en grande partie sur des méthodes cryptographiques par la conception de protocoles *cryptographiques* [7] de sécurité. D'autre part, comme il est difficile de sécuriser un réseau sans commencer, d'abord, par y contrôler l'accès, les protocoles d'authentification sont parmi les protocoles cryptographiques qui ont été les plus étudiés.

Au départ, l'authentification était envisagée entre un client souhaitant accéder à certaines ressources et un serveur d'authentification protégeant ces ressources. Les protocoles d'authentification étaient alors assez simples de conception [8]. Mais au fur et à mesure que la taille des réseaux et le nombre de clients augmentaient d'une part et que les services se diversifiaient d'autre part, les parties impliquées dans un processus d'authentification se sont multipliées [9] et les protocoles d'authentification sont devenus plus complexes et sophistiqués [10]. Cette évolution s'est aussi accompagnée par l'ajout de deux autres fonctions au serveur d'authentification qui sont la fonction d'autorisation et la fonction de collecte pour les besoins de comptabilité. C'est ainsi que ce serveur est devenu un serveur AAA et l'on parle alors d'architecture AAA (*Authentication, Authorization, Accounting*). Mais, outre les entités intermédiaires impliquées dans l'authentification, quand cela est nécessaire, une architecture AAA peut com-

5. « Toute personne, en tant que membre de la société, a droit à la sécurité sociale ; elle est fondée à obtenir la satisfaction des droits économiques, sociaux et culturels indispensables à sa dignité et au libre développement de sa personnalité, grâce à l'effort national et à la coopération internationale, compte tenu de l'organisation et des ressources de chaque pays. »Disponible sur `http://www.un.org/fr/documents/udhr/index.shtml`

6. Les besoins les plus importants étant ceux les plus près de la base dans la pyramide de Maslow, les *besoins de sécurité* du corps, de l'emploi, de la santé, de la propriété, etc. viennent juste au dessus des premiers besoins vitaux c'est-à-dire les *besoins physiologiques*, à savoir manger, boire, dormir, respirer, etc.

7. Un protocole cryptographique est un protocole qui utilise des algorithmes cryptographiques, notamment des algorithmes de chiffrement, de hachage et/ou de signature de qualité suffisante.

8. Les exemples de tels protocoles incluent PAP (*Password Authentication Protocol*) et CHAP (*Challenge Handshake Authentication Protocol*).

9. L'ensemble de ces parties inclut un client AAA, un serveur AAA, un ou plusieurs proxies AAA, des agents AAA, etc.

10. Le protocole EAP (*Extensible Authentication Protocol*) combine plusieurs méthodes cryptographiques et repose, dans certains scénarios, sur des protocoles cryptographiques complémentaires, combinant ainsi les apports, en terme de sécurité, de chacun des deux protocoles.

porter des serveurs spécialisés dans la tâche de collecte et des bases de données contenant les informations nécessaires à l'authentification et l'autorisation des utilisateurs.

Il est vrai que le remplacement des liaisons filaires par des liaisons sans-fil au niveau de la boucle locale a introduit de nouveaux problèmes de sécurité qui ont été relativement faciles à résoudre et à prendre en charge dans la conception des architectures AAA. Par contre, lorsque les utilisateurs deviennent mobiles, que ce soit dans un réseau cellulaire ou Mobile IP, la sécurité devient un problème épineux du fait du trafic de signalisation supplémentaire à protéger et de la gestion d'itinérance qui implique une gestion de clés plus complexe par l'architecture AAA.

Cela est d'autant plus vrai dans les réseaux ad hoc où toute administration centrale est à priori absente et où, par conséquent, les opérations de création et de maintien du réseau sont partagées entre tous ses nœuds provoquant ainsi un trafic de signalisation d'un volume plus important que celui d'un réseau à infrastructure. Ce volume dépend, en plus, de la dynamicité de la topologie du réseau, elle même fonction de la mobilité des nœuds : plus la mobilité est grande, plus ce volume est important. D'autre part, de l'absence d'infrastructure résulte, aussi, que toute sorte de trafic, en particulier le trafic de signalisation, se trouve emprunter des chemins inconnus. Toutes ces raisons font que les problèmes de sécurité sont plus délicats à gérer et qu'en compensation, les moyens afin de les résoudre doivent être plus importants et les solutions plus perfectionnées.

Plusieurs propositions ont tenté d'apporter un remède en se basant, en grande partie, sur la présence d'un opérateur. Ainsi, l'architecture AAA présente dans le réseau de celui-ci voit ses capacités étendues au réseau ad hoc formé à proximité de ce réseau. Mais la réconciliation de la mobilité, de l'absence d'infrastructure et de la sécurité reste un objectif imparfaitement atteint dans un tel type d'organisation.

## Organisation du mémoire

Le mémoire s'articule autour de trois parties. La première partie est composée de trois chapitres. Dans le premier, nous nous attachons à donner une définition claire des réseaux ad hoc et nous présentons leurs caractéristiques, leurs domaines d'application et leurs défis afin de mieux comprendre les problèmes de sécurité dans ce type de réseau. Dans le deuxième, nous exposons les méthodes et les outils de sécurité à notre disposition dans les réseaux à infrastructure : d'abord les techniques de contrôle d'accès, ensuite les algorithmes cryptographiques, puis les moyens de gestion de clés et d'établissement de relation de confiance, enfin quelques protocoles de sécurité. Dans le troisième chapitre, nous passons en revue les solutions AAA proposées pour les réseaux ad hoc et plus particulièrement les MANETs. Nous en faisons une taxonomie et nous mettons l'accent sur les aspects qui n'ont pas été suffisamment abordés dans ces solutions, surtout dans le cas où la présence d'un opérateur n'est pas forcément permanente.

La deuxième partie comporte deux chapitres. Le quatrième chapitre présente la solution que nous proposons afin de remédier au problème de présence intermittente d'une architecture AAA d'un opérateur à proximité d'un MANET. Cette solution obéit à un cahier des charges que nous avons établi. Celui-ci met

l'accent sur la nécessité de concevoir non seulement une architecture AAA *distribuée* dans ce cadre mais aussi des protocoles AAA distribués. Le cinquième chapitre évalue certains aspects de la sécurité de la solution proposée au moyen de l'outil de validation automatique des protocoles de sécurité AVISPA.

La troisième partie porte sur l'évaluation du trafic engendré par le protocole AAA que nous avons défini en ayant recours aux simulations avec ns-2. Dans le sixième chapitre, nous traitons un exemple simple avec un réseau ad hoc formé de nœuds immobiles disposés en étoile. Dans le septième chapitre, nous étudions un cas plus complexe où le MANET considéré comporte une centaine de nœuds et où plusieurs paramètres de "mobilité" et de "distribution" varient.

# Première partie

# Réseaux ad hoc mobiles et exigences des protocoles AAA : un tour de l'existant

# Chapitre 1

# Réseaux ad hoc et problèmes de sécurité

## Sommaire


**1.1 Introduction** . . . . . **3**
**1.2 Aperçu** . . . . . **4**
1.2.1 Définition et vocabulaire . . . . . 4
1.2.2 Paradigme . . . . . 5
1.2.3 Le routage : un problème fondamental . . . . . 7
**1.3 Applications : entre passé et futur** . . . . . **9**
1.3.1 Le départ : les militaires . . . . . 9
1.3.2 Évolution : les civiles . . . . . 12
1.3.3 Civiles non commerciales . . . . . 14
1.3.3.1 Secours . . . . . 14
1.3.3.2 Autres . . . . . 14
1.3.4 Civiles commerciales . . . . . 15
**1.4 Défis** . . . . . **17**
1.4.1 Liens sans-fil . . . . . 17
1.4.2 Réseau multi-sauts et nœuds mobiles . . . . . 18
1.4.3 Absence d'une administration centrale . . . . . 18
1.4.3.1 Absence de relations de confiance . . . . . 19
1.4.4 Contexte distribué . . . . . 19
1.4.5 Nœuds égoïstes . . . . . 19
1.4.6 Nœuds malveillants . . . . . 19
1.4.7 Contrainte d'énergie . . . . . 20
1.4.8 Terminaux hétérogènes . . . . . 20
**1.5 Conclusion** . . . . . **20**


## 1.1 Introduction

L'étude des réseaux ad hoc prend une place de plus en plus importante dans la recherche et tout semble indiquer qu'ils prendront bientôt une part

importante dans notre vie quotidienne (cf. section 1.3.2). Par leur flexibilité et l'économie engendrée en matière de déploiement de réseaux, leur intérêt dépasse les domaines purement militaire ou d'opérations de secours. Depuis les années 1990 et l'essor des réseaux personnels et locaux sans-fil, cet intérêt couvre aussi le domaine publique à travers un catalogue d'applications diverses. Mais certains problèmes inhérents à ces réseaux impliquent une conception complexe de ces applications et surtout de celles qui nécessitent une extension du réseau Internet.

Dans ce chapitre, nous donnons d'abord un aperçu des réseaux ad hoc, de leur paradigme et du fonctionnement du routage en leur sein. Nous présentons ensuite brièvement leur évolution depuis les années 1970 et plus particulièrement l'évolution du champ des applications qu'ils couvrent. Enfin, nous investissons les défis qui vont à l'encontre du développement des applications publiques. Nous verrons qu'une partie importante d'entre eux sont en fait des problèmes de sécurité.

## 1.2 Aperçu

### 1.2.1 Définition et vocabulaire

Un réseau ad hoc c'est à dire selon le Trésor de la Langue Française informatisé (TLFi) [3], *spécialement institué pour un objet précis auquel il convient parfaitement*, désigne en fait un réseau d'équipements terminaux, par exemple des ordinateurs portables, mais aussi peut-être des téléphones cellulaires, des PDAs, des capteurs, des instruments de mesures, communiquant entre-eux par des ondes électromagnétiques dont la fréquence est le plus souvent de l'ordre du gigahertz [1], on parle alors de liaisons sans fil.

Les réseaux ad hoc [4, 5] sont des réseaux sans infrastructure c'est à dire qu'il n'y a pas de routeurs séparés pour aiguiller le trafic d'une source vers une destination et qu'il n'y a pas non plus d'entités centrales pour administrer le réseau contrairement à ce qui est la règle dans le cas des réseaux filaires et cellulaires. La constitution du réseau et le maintien de son fonctionnement sont assurés par les terminaux eux-mêmes. On parle de réseau *autonome* déployé de façon *spontanée*.

Dans un tel réseau, on appelle les terminaux *terminaux ad hoc*. Un terminal ad hoc est ainsi doté d'une ou plusieurs interfaces réseau permettant la mise en œuvre d'une technologie sans-fil telle que le Bluetooth pour des liaisons à très courte portée [2], le WIFI pour des liaisons à courte portée [3] ou le WIMAXpour des liaisons à portée un peu plus longue [4]. Chaque interface est dotée

1. Les principales technologies sans-fil utilisées dans les réseaux ad hoc mettent en œuvre des signaux de fréquences hertziennes. Toutefois, la lumière infrarouge peut être utile pour des réseaux de courte distance (de l'ordre du mètre), connus sous le nom de *PANs* (*Personnel Area Networks*), entre appareils communiquant grâce à la norme *IrDA* (*Infrared Data Association*).

2. De quelques mètres pour une puissance de 1 mW, à une dizaine de mètres pour 2,5 mW, ou à une centaine de mètres pour 100 mW.

3. D'une dizaine de mètres selon la norme IEEE 802.11a jusqu'à quelques centaines de mètres selon les normes IEEE 802.11b, 802.11g et 802.11n en extérieur. Mais restreinte à quelques dizaines de mètres à l'intérieur des bâtiments.

4. Selon le milieu, urbain ou rural, la distance est entre 1km et une quinzaine de kilomètres étant entendu que lorsque celle-ci augmente, le débit maximum diminue.

d'une antenne[5] qui peut recevoir ou émettre également dans toutes les directions permettant ainsi une diffusion dans le voisinage, on parle alors d'*antenne omnidirectionnelle*. Elle peut aussi être dotée d'une antenne émettant de façon privilégiée dans un angle solide particulier, on parle alors d'*antenne directionnelle*. Ce sont ces dernières que l'on utilise pour mettre en œuvre des liaisons point à point[6].

Théoriquement, il suffit qu'au moins deux terminaux ad hoc soient à la portée l'un de l'autre pour pouvoir constituer un réseau ad hoc. En général les terminaux ne sont pas en portée directe i.e. qu'ils ne peuvent pas tous communiquer directement entre eux. On dit qu'ils ne sont pas en portée directe. Dès lors, ils peuvent être amenés à relayer les paquets pour le compte d'autres terminaux ou à faire relayer leurs propres paquets ce qui permet de les faire parvenir aux destinations désignées. Cette organisation est essentielle au concept même de réseau ad hoc. Un terminal se comporte ainsi comme un routeur capable de s'attacher à plusieurs autres terminaux pouvant d'ailleurs avoir chacun une ou plusieurs interfaces d'une technologie sans-fil différente. Il peut être ainsi lié *via* une technologie A à un premier terminal et grâce à une technologie B différente à un deuxième terminal. En tant que routeur, il achemine du trafic pour le compte de certains terminaux, les *sources*, vers d'autres terminaux, les *destinations*. Éléments de cette chaîne de transmission, on le désigne pour cela par le terme *relais*. Mais il est aussi capable d'envoyer et de recevoir du trafic pour son propre compte.

On modélise un réseau ad hoc par un graphe où les terminaux sont les nœuds du graphe qu'on appelle donc *nœuds ad hoc*. Deux nœuds sont reliés par une arête, on dit qu'ils sont *à portée* l'un de l'autre, lorsque les équipements qu'ils modélisent sont susceptibles d'établir une liaison ce qui suppose des conditions de distance, de puissance émise, de sensibilité de réception et de niveau de bruit. On voit qu'une arête symbolise le fait que le trafic d'un nœud peut atteindre directement l'autre nœud.

### 1.2.2 Paradigme

Un des facteurs importants dans la classification des réseaux ad hoc est l'étendue qu'ils peuvent couvrir. Elle dépend de la portée du signal sans-fil donc de la technologie sans-fil sous-jacente. On trouve ainsi les réseaux appelés PANs (*Personal Area Networks*) où les technologies comme le Bluetooth et l'IRdA peuvent être utilisés sur quelques mètres. Ensuite les LANs (*Local Area Networks*) liés essentiellement à la technologie WIFI portant sur quelques centaines de mètres. Ils incluent aussi les PANs. Enfin les MANs (*Metropolitan* Area Networks) mettant en œuvre par exemple le WIMAX peuvent porter sur quelques kilomètres. Ils incluent les PANs et les LANs. Notons au passage que cette classification correspond à celle que l'on connaît dans les réseaux filaires. Elle peut sans doute inclure des étendues plus grandes, comme les WAN (*Wide Area Networks*), mais nous nous n'en parlerons pas ici. Quoiqu'il en soit, nous

5. Certaines technologies utilisent deux antennes pour une même interface. La gestion des déphasages induits permet une augmentation de la portée.

6. A l'émission, une antenne directionnelle concentre l'essentiel de la puissance émise dans l'angle solide de référence et permet donc une meilleure gestion de la puissance. En réception, ce type d'antenne favorise les signaux provenant des directions de référence limitant ainsi la réception des signaux parasites tels que les bruits atmosphériques ou industriels.

nous intéressons dans cette thèse aux LANs et aux applications qui peuvent s'y rattacher [7].

Dans ces conditions, comme un réseau ad hoc se forme rapidement dans un but précis, chaque nœud le constituant est en mesure de s'adapter aux conditions extérieures liées à un contexte [8] donné. La nature des terminaux, le rôle qu'ils

| déclinaisons / Ad hoc | terminaux | rôles | répartition | mouvement | dépendance |
|---|---|---|---|---|---|
| MANET | hétérogènes | symétriques | aléatoire | aléatoire | faible |
| WSN | homogènes | symétriques | aléatoire | limité | forte |
| VANET / InVANET | homogènes | symétriques | rectiligne | rapide | forte |
| Tactiques militaires / Situations d'urgence | très hétérogènes | asymétriques | aléatoire | aléatoire | très forte |
| Maillés | très hétérogènes | asymétrique | aléatoire | fixe / aléatoire | forte |

Tableau 1.1 – Diverses déclinaisons de réseaux ad hoc

jouent, leur répartition dans l'espace, la présence ou non de mouvements et la dépendance éventuelle d'un réseau à infrastructures sont quelques uns parmi les éléments qui, une fois fixés, définissent un contexte de travail et amènent à décliner les réseaux ad hoc en plusieurs catégories. Le tableau 1.1, que nous avons établi d'après [6], montre cinq types de réseaux ad hoc, que l'on rencontre souvent décrits dans la littérature ou réalisés dans l'industrie.

Lorsque un réseau constitué d'ordinateurs portables, de téléphones portables, de PDA est hétérogène, lorsque ces terminaux sont pour certains susceptibles de se déplacer sur une surface ou dans l'espace, on utilise dans le monde anglo-saxon la désignation *Mobile Ad hoc Network* (réseau ad hoc mobiles), souvent abrégée en MANET, mettant ainsi l'emphase sur la mobilité. Lorsque ce réseau est constitué de capteurs ne bougeant quasiment jamais et servant à mesurer des données géolocalisées (paramètres chimiques, climatiques, sismiques, micro-déplacements), des données médicales ou toutes autres données environnementales [9], on parle de réseaux de capteurs sans-fil (en anglais *Wireless Senser Networks* abrégé en WSN [10]). C'est la classification qui a été adoptée par le NIST (*National Institute of Standards and Technology*) [7] depuis 2001.

Parmi les MANETs, on identifie les réseaux de véhicules équipés de liaisons sans-fil se déplaçant à assez haute vitesse (quelques dizaines de m/s) selon une

7. La norme IEEE 802.11 du WiFi sera utilisée ultérieurement dans les simulations. Nous nous intéresserons donc pas à la norme IEEE 802.15 du Bluetooth.

8. Un contexte est l'ensemble des caractéristiques décrivant un réseau mobile et son environnement.

9. Les réseaux de capteurs sont aussi utiles dans le domaine militaire afin par exemple de détecter les intrusions et de localiser des combattants, des véhicules ou des armes. Ils auront aussi un intérêt dans le domaine commercial pour améliorer le processus de stockage et de livraison en contrôlant par exemple la chaîne de froid.

10. Contrairement aux MANETs, ces réseaux, faisant plutôt partie des réseaux PANs, sont contraints par des capacités limités en énergie car son renouvellement sur le terrain est problématique. Un recourt à des énergies renouvelables comme l'énergie solaire est en cours d'étude et d'expérimentation. En plus, les WSN sont contraints par une sensibilité aux pannes. Enfin, le nombre de capteurs dans un WSN est beaucoup plus important que le nombre de nœuds dans un MANET. La conception de solutions de sécurité pour ces réseaux ne doit pas négligée ces aspects.

trajectoire quasiment rectiligne le long d'une voie de communication, on parle alors de *Vehicular Ad hoc NETworks* ou VANET. Les terminaux sont alors, outre leurs liaisons réciproques, en communication avec un réseau à infrastructure géré par un opérateur avec lequel ils communiquent par le biais de relais disposés à intervalles plus ou moins réguliers le long de la voie de communication.

Mais l'on peut rencontrer des réseaux hybrides combinant des MANETs et des WSNs en plus éventuellement d'un réseau central. Les réseaux tactiques militaires et les réseaux déployés en cas de situation d'urgence incluent, en plus des terminaux portables légers organisés en MANETs, comme ceux que nous avons décrits ci-dessus, des équipements ayant des capacités plus grandes en termes de puissance de calcul ou de capacité de stockage pouvant leur conférer un rôle plus important que les autres terminaux, par exemple celui de relais vers un réseau central de gestion et de commandement. Enfin les réseaux maillés [11], qui peuvent aussi dépendre d'un réseau d'opérateur, servent à étendre celui-ci par le déploiement de points d'accès fixes. Tout autour de ceux-ci, des clients mobiles, organisés en MANET, peuvent accéder au réseau de cet opérateur ou même à Internet grâce à la médiation ainsi mise en place.

Dans cette thèse, nous focaliseront nos travaux sur les MANETs. Ainsi nous examinerons les aspects de sécurité et par la suite les fonctions AAA en rapport avec les applications qui peuvent être supportées par un MANET.

### 1.2.3 Le routage : un problème fondamental

Le développement des MANETs a eu lieu, en partie, dans le cadre de l'unification des supports de transport de services en réseau dans le but d'utiliser un protocole commun qui avait fait preuve de sa robustesse, de son efficacité et de sa capacité à passer à l'échelle en révolutionnant le monde informatique avec la diffusion d'Internet. Il s'agit du protocole IP (*Internet Protocol*). C'est donc la technologie qui a été choisie pour transporter toutes sortes de données dans les MANETs.

Par ailleurs, cet objectif s'est inscrit et s'intègre encore dans l'effort de fournir de plus en plus de services en mobilité. Pour cette raison, l'ambition du groupe MANET à l'IETF (cf. ci-dessous) était de concevoir le routage comme un protocole pair-à-pair [12] (*Peer to Peer Networks*) entre nœuds mobiles communiquant avec des technologies sans-fil. De ce fait et puisque les nœuds jouent des rôles symétriques en étant tous routeurs, le contexte des MANETs est considéré comme *distribué* ou *réparti* et le rouage comme étant *multi-sauts*. Selon Tanenbaum « un système réparti est un logiciel élaboré au-dessus d'un réseau pour apporter un haut degré de cohésion et de transparence ». Le routage dans les MANETs concourt, à l'intérieur de la couche réseau, à l'abstraction des couches inférieures. Il joue en cela le rôle d'une sorte de colle entre les diverses technologies sans-fil ainsi agrégées.

Le routage se révèle donc fondamental dans l'architecture des MANETs. C'est pour cette raison que le groupe MANET à l'IETF a été fondé en ayant

11. Des solutions commercialisées de réseaux maillés existent déjà comme celle proposée par Motorola et appelée MeshNetworks [8] ou celle conçue par SPANworks [9].

12. Un réseau pair-à-pair, dans sa définition la plus stricte, est formé d'ordinateurs jouant chacun, en même temps, le rôle de client, pouvant demander des fichiers distants, et de serveur, fournissant des fichiers aux autres clients demandeurs. Il s'agit de réseau virtuel reposant sur l'architecture matérielle et logicielle d'Internet.

comme tâche d'élaborer des standards de protocoles de routage dont nous donnons un rapide aperçu dans cette section. Le premier RFC publié par ce groupe, le RFC 2501, met l'accent sur les critères qualitatifs et quantitatifs à suivre lors de la conception et de l'évaluation d'un protocole. Nous présentons quelques uns parmi eux à titre indicatif et, aussi, parce que nous pensons qu'une partie d'entre eux reste valable pour évaluer d'autre protocoles au sein des MANETs.

Comme dans un réseau à infrastructure, le rôle d'un protocole de routage est de construire, le plus souvent à la volée, l'ensemble des routes nécessaires à acheminer le trafic entre les nœuds. Cela se traduit par le remplissage de tables de routage dont les entrées précisent à quel routeur un trafic doit être envoyé pour lui permettre d'atteindre une destination donnée. La définition d'un mécanisme de découverte de routes est donc un problème de base dans la conception de protocole de routage.

Le caractère distribué du mécanisme de découverte des routes est un aspect fondamental. S'y ajoute un certain nombre de contraintes qualitatives : les routes ne doivent pas contenir des boucles ; leur découverte doit, de référence et quand les conditions le permettent, tenir compte des besoins des nœuds en nouveaux chemins au lieu de considérer que ce besoin est uniforme dans l'espace et dans le temps [13] ; les problèmes de sécurité comme l'espionnage des paquets de routage, leur modification, leur rejeu (cf. section 2.4.3.5 du chapitre 2), et la re-direction doivent être pris en compte ; enfin, des périodes aléatoires de mise en veille d'un terminal doivent être possibles à des fins d'économie d'énergie (cf. plus bas aussi)

Quant aux critères quantitatifs, ils préconisent de mesurer les performances et le rendement de ces protocoles selon des métriques incluant le volume de données de routage échangées, le délai de bout en bout, le temps de découverte d'une route et le pourcentage de paquets délivrés dans l'ordre de leur envoi. Les performances doivent être mesurées compte tenu du trafic de contrôle provenant des autres couches du réseau comme la couche de liaison de données. Ceci est important à évaluer si le même canal est partagé entre diverses sortes de trafic. La taille du réseau en nombre de nœuds est aussi à prendre en compte, tout comme la *connectivité* du réseau i.e. le degré moyen d'un nœud [14], le taux de changement de la topologie [15], la capacité des liens [16], le pourcentage de liens unidirectionnels [17], le modèle de trafic [18], la mobilité [19], la fraction de nœuds en mode veille [20] , etc.

Les protocoles ayant fait l'objet de RFC de catégorie expérimentale (en an-

13. Ainsi les ressources d'énergie et en bande passante sont mieux gérées mais cela au prix d'opérations de routage plus longues à cause du temps passé à la découverte de routes au moment où une destination est demandée.

14. Le degré d'un nœud dans un graphe est le nombre de ses voisins.

15. C'est la vitesse à laquelle la topologie du réseau change. Alors que la géométrie d'un graphe s'occupe de définir la position des nœuds dans un repère et l'ensemble des arrêtes qui les relient, la topologie s'attache à évaluer le nombre de voisins d'un nœud, la connexité du graphe et des notions apparentées.

16. Le débit effectif d'un lien après comptabilisation des pertes due au aux multiples accès, au codage, au cadrage (en anglais *framing*).

17. Cela afin d'observer le comportement du protocole en présence de liens unidirectionnel.

18. Cela afin de mesurer la capacité d'adaptation du protocole à des modèles de trafic non uniforme ou en rafales.

19. Quand et sous quelles conditions la corrélation topologique temporelle et spatiale est-elle importante dans l'évaluation d'un protocole de routage ? Dans ces cas là, quel est le modèle approprié pour simuler la mobilité des nœuds dans les MANETs ?

20. Cela afin de mesurer les performances du protocole en présence de nœuds en veille ou pas.

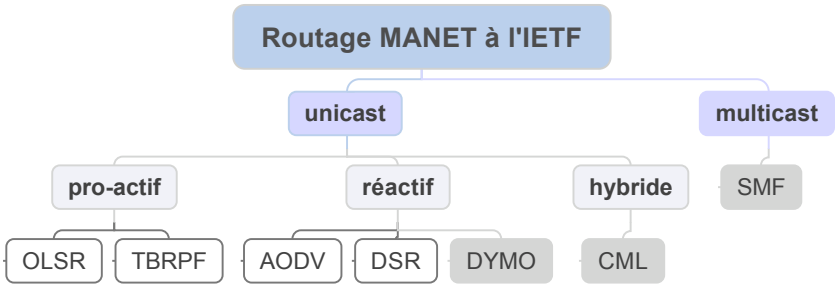


FIGURE 1.1 – Protocoles de routages pour les MANET définis au sein de l'IETF

glais *Experimental*) sont à ce jour au nombre de quatre. Il n'empêche que de nombreux autres sont en cours de propositions sous forme de *drafts*[21]. La figure 1.1 montre que les protocoles de routage sont classés en famille. D'abord, les *unicasts*[22] sont ceux qui routent le trafic entre exclusivement deux nœuds. Ils sont au cœur des propositions. Ensuite, les *multicasts* sont responsable de délivrer un trafic au sein d'un groupe de nœuds. Il existe trois types de routage *unicast* : les , les *réactifs* et les *hybrides*, une sorte de mélange des deux premiers. OLSR (*Optimized Link State Routing*) [10] et TBRPF (*Topology Dissemination Based on Reverse-Path Forwarding*) [11] font partie des protocoles pro-actifs, alors que AODV (*Ad hoc On-Demand Distance Vector*) [12] et DSR (*Dynamic Source Routing*) [13] sont de type réactif. DYMO (*Dynamic MANET On-demand*) [14] appartient aussi à cette dernière catégorie et il a fait dernièrement l'objet de plusieurs drafts. CML (*ChaMeLeon*) [15] est un protocole hybride faisant partie actuellement des documents actif à l'IETF sans pour autant être une publication du groupe MANET. Enfin, SMF (*Simplified Multicast Forwarding*) [16] est un protocole *multicast* qui a été analysé dernièrement dans plusieurs drafts[23].

## 1.3 Applications : entre passé et futur

### 1.3.1 Le départ : les militaires

Les militaires ont été parmi les premiers à utiliser les réseaux ad hoc mobiles dans le cadre de leur réseaux tactiques afin d'améliorer la qualité et la durée des communications pendant les opérations. Ces dernières étant dynamiques, il est en général difficile de compter sur l'existence d'une infrastructure de réseau pré-établie sur le terrain et l'on y privilégie les liaisons sans-fil. Mais, à cause des phénomènes physiques de réflexion, de réfraction et d'interférences, un si-

21. Avant d'être acceptée et de passer sous forme de RFC (*Request For Comments*), une spécification est soumise à l'IETF sous forme de *drafts* en plusieurs versions successives demandant des révisions.

22. Les protocoles *unicast* sont ceux sur lesquels se concentrent les travaux du groupe MANET

23. Une liste exhaustive de protocoles de routage pour les MANETs est consultable sur la page `http://en.wikipedia.org/wiki/List_of_ad_hoc_routing_protocols`. On note que plusieurs d'entre eux ont été proposés en dehors de l'IETF et que certains ont été abandonnés après avoir déçus les espoirs dont ils étaient porteurs.

gnal radio ne peut pas se propager au delà de la ligne d'horizon (en anglais *line of sight* ou LOS) [24], ce qui rend les communications sans-fil impossibles au delà de cette limite. C'est pourquoi l'emploi des réseaux ad hoc mobiles s'était avéré intéressant dans la mesure où non seulement ils ne se basent pas sur une infrastructure pré-existante mais aussi leurs nœuds permettent de relayer eux-même les communications permettant ainsi de s'affranchir de la limite physique de la ligne d'horizon [1]. En fait, on peut même imaginer une intégration du nœud dans la tenue [25] de combat, le réseau s'identifiant alors à l'unité combattante en opérations.

Aux États Unis d'Amériques, les premières applications militaires utilisant les MANETs remontent au lancement par la DARPA (*Defense Advanced Research Projects Agency*) [17] du projet PRNet [18] (*Packet Radio Network*) en 1972. L'un de ces objectifs était d'étudier l'adaptation des protocoles d'ARPANET, le précurseur d'Internet, notamment les techniques de commutation par paquets [26] à l'environnement sans-fil et mobile des radios. C'est ainsi que l'architecture distribuée d'un réseau PRNet mettait en œuvre des radios [27] utilisant un routage en mode différé pour étendre la couverture du réseau sur une large zone géographique.

En 1983, la DARPA a lancé le développement des réseaux SURAN (*Survivable Radio Networks*) en vue de concevoir des radios de plus petite taille, moins chers, consommant moins d'énergie et supportant des protocoles de radiocommunication par paquets plus complexes que que ceux des réseaux PRNet. On y a, entre autre, examiné des problèmes liés au passage à l'échelle (de 10 à 1000 nœuds) à travers la clusterisation (en anglais *clustering*) [28] dynamique et des problèmes liés à la sécurité, la puissance de traitement et la gestion d'énergie. Cet effort a aboutit à l'invention du LPR (*Low-cost Packet Radio*) [19] en 1987 qui utilisait la technique DSSS (Direct Sequece Spread Spectrum) et intégrait un microprocesseur Intel 8086 basé sur la commutation par paquets.

A la fin des années 1980 et au début des années 1990, ces premières idées de réseaux de radios à commutation par paquets se sont révélées encore plus pertinentes et plus aisément réalisables avec la révolution due au micro-ordinateur et aussi aux premiers ordinateurs portables équipés de cartes sans-fil et de ports infrarouge. Ainsi, pour en tirer profit dans l'environnement sans-fil et mobile, le DoD (*Department of Defense*) lança en 1994 le programme de DARPA appelé GloMo (*Global Mobile*) *Information Systems* [20]. Ce programme visait le développement de réseaux ad hoc mobiles et en particuliers d'appareils sans-

24. Quand il est émis dans les fréquences supérieures à 100 MHz

25. En France, Sagem a proposé un gilet se portant au dessus du gilet par balles qui comprend l'électronique dont un soldat a besoin : localisation GPS, prises de photos et de vidéos, etc.

26. Le partage de bande passante et le routage en mode différé faisaient partie de ces techniques. En mode différé (en anglais *Store and Forward*), un commutateur attend d'avoir complètement reçu et analysé une trame de données sur l'une de ses interfaces d'entrée pour commencer sa retransmission sur une ou plusieurs de ses interfaces de sortie. Ce mode impose la mise en mémoire tampon de la totalité de la trame et retarde sa transmission. En contrepartie, cela garantit l'intégrité de la trame et évite les erreurs. Cela implique aussi un délai de commutation plus important.

27. Les radios d'un réseau PRNet mettaient en œuvre une combinaison de deux protocoles d'accès au canal, CSMA et Aloha (*Areal Locations of Hazardous Atmospheres, université Hawaï*) afin de pouvoir partager le canal radio en diffusion.

28. La clusterisation est le processus de formation de *cluster* c.-à-d. de groupes d'équipements proches les uns des autres dans l'espace et bougeant ensemble.

FIGURE 1.2 – L'Internet tactique. *Source : Thalès*

fil supportant une connectivité multimédia de type Ethernet à tout moment et n'importe où. Plusieurs conceptions de réseaux ont été alors explorées, par exemple un réseau d'architecture plane pair-à-pair appelé WINGs (*Wireless Internet Gateways*) et proposé par l'UCSC (*University of California, Santa Cruz*), ou aussi un réseau d'architecture hiérarchique basée sur la clusterisation appelé MMWN (*Multimedia Mobile Wireless Network*) et mis en place par GTE Internetworking [29].

En 1997, l'armée américaine a mis en place l'IT (*Internet Tactique*) [30] (cf. FIGURE 1.2), qui a permis la plus grande implémentation à large échelle des réseau mobile, sans-fil et multi-sauts de radios à commutation par paquets. En effet, la plate-forme *Force XXI Battlefield Command Brigade and Below* ou FBCB2 [21] basée sur MANET a été mise en place afin de permettre aux soldats de localiser les forces alliées et adverses sur le champ de bataille. La technique DSSS (Direct Sequence Spread Spectrum) au niveau physique, le protocole TDMA (Time Division Multiple Access) avec un débit de l'ordre de 10 Kbps au niveau liaison de données, et des modifications de protocoles Internet jusque là utilisés dans un cadre commercial y ont été testés. Cet expérience a démontré que les protocoles en vogue dans les réseaux filaires commerciaux n'étaient pas forcément efficaces dans le contexte des réseaux sans-fil à topologie dynamique, à débit bas et ayant un taux d'erreur élevé.

En 1999, un autre déploiement des MANETs dans le cadre de ELB ACTD (*Extending the Littoral Battle-space Advanced Concept Technology Demonstration*) a été exploré afin de montrer la faisabilité de certains concepts de conduite de la guerre imaginés par les corps de la Marine (*Marine Corps*). Les scénarios mettaient en œuvre des communications au delà de l'horizon (*over-the-horizon*), à travers un relais aérien, entre les navires militaires en mer et les soldats sur terre. Le réseau était formé d'une vingtaine de nœuds et d'appareils WaveLAN

29. *General Telephone & Electronics Corporation* était le produit de la fusion des deux entreprises *General Telephone* et *Sylvania Electric Products* en 1959. Elle était alors devenue leader en matière d'éclairage, de télévision, de radio, de chimie et de métallurgie. Elle a acheté plusieurs autres entreprises depuis jusqu'à sa fusion en 2000 avec l'entreprise *Bell Atlantic* pour devenir *Verizon Communications*.

30. L'une des priorités de l'armée dans la numérisation de l'espace de combat est de distribuer des capacités de commandement et de contrôle au travers de l'ensemble des forces. Cela nécessite un réseau intégrant notamment les systèmes de combat et les unités déployées (intégration horizontale) et prenant en charge efficacement le cycle de contrôle de décision et de commandement (intégration verticale) assurant ainsi une réelle supériorité. Pour atteindre ce but, un tel réseau doit permettre d'établir une connectivité et des communications fiables, transparentes, et sûres pour tous les membres de l'armée. L'Internet tactique est le terme utilisé pour décrire ce réseau de communications intégré dans l'espace de combat. Ce terme a été choisi en raison de similitudes fonctionnelles avec l'Internet commercial.

et VRC-99A fabriqué par Lucent et employés dans le cœur du réseau ou comme équipements d'accès. ELB ACTD a ainsi réussi à montrer le succès des relais à connecter des utilisateurs au delà de la ligne de mire. Les applications militaires utilisant les réseaux MANETs continuent à faire l'objet de projets à la DARPA notamment à travers le projet ITMANET (*Information Theory for Mobile Ad hoc Networks*) qui a commencé en 2007 et finira en 2011 et dont l'objet est de développer de puissantes technologies [22].

En Europe, l'IT mobile et le développement des MANETs qui s'en était suivi a commencé au début des années 2000 [23] en profitant des avancées américaines en la matière [24]. L'entreprise Thalès est l'un des leaders dans ce domaine à travers le projets M@tis (*Mobile Tactical Internet*) [25].

### 1.3.2 Évolution : les civiles

Aux USA, au milieu des années 1990, avec la définition du standard IEEE 802.11 [26] qui permettait de mettre en place des réseaux locaux sans-fil que l'on dénommait déjà « réseaux ad hoc »au sein de l'IEEE (*Institute of Electrical and Electronics Engineers*), des technologies radio commercialisables ont commencé à apparaître sur le marché. Depuis, la communauté de chercheurs dans le domaine des réseaux sans-fil a pris conscience du potentiel commercial important et des avantages des réseaux mobiles ad hoc en dehors du contexte militaire. En 1996, un standard européen concurrent appelé HIPERLAN (*HIgh PERformance radio LAN*) [27] et mettant l'accent sur le routage ad hoc avait été ratifié par l'ETSI (*European Telecommunications Standards Institute*). Mais contrairement au standard IEEE 802.11, celui-ci n'a pas été suffisamment soutenu par les acteurs économiques [31] notamment en raison de son utilisation de la bande des 5GHz où opéraient les satellites et les miliaires [32]. Par ailleurs, la fondation du groupe MANET en 1998 au sein de l'IETF (*Internet Engineering Task Force*) avait pour but de standardiser des protocoles de routage [5] pour les MANETs suite aux développements acquis au cours du projet GloMo.

L'importance de la recherche tout comme celle de l'effort de normalisation concernant les MANETs peut être constatée par une étude statistique des documents issus des grands organismes de normalisation. Par exemple, au milieu de l'année 2010, à l'IETF, les *drafts* se rapportant aux MANETs étaient au nombre de 24 drafts sur un total de 1657 drats, soit 1,45% des drafts soumis, et les RFCs étaient au nombre de 11 sur un total de 5754 RFCs environ, soit 0,2% des RFCs produits par l'IETF. En ce qui concerne l'IEEE, nous avons étudié entre 1995 et 2010 l'évolution du nombre d'articles soumis ayant pour objet les MANETs d'une part et la norme IEEE 802.11 d'autre part. Les courbes illustrées par la figure 1.3 montrent que la majeure partie des papiers traitant de la norme 802.11 a pour objet les MANETs. Elles présentent, par ailleurs, une évolution quasiment exponentielle depuis 2002 ce qui montre l'intérêt croissant que suscitent ce type de réseaux.

Cet intérêt pour l'existant s'est accompagné par l'invention de toutes sortes

31. L'entreprise Ericsson a choisit de se retirer en 2002 du projet BRAN (*Broadband Radio Access Networks*) qui développait cette norme, voyant dans la norme IEEE 802.11a un meilleure opportunité pour vendre ses produits.

32. L'évolution de la norme HIPERLAN a continué toutefois par la définition de 4 versions depuis. Il est prévu qu'elle face partie des technologie sans-fil employées dans le réseau de 4$^{\text{ème}}$ génération.

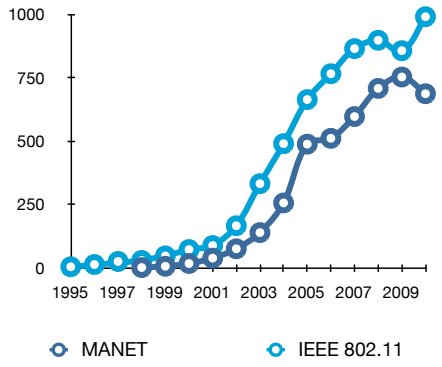


FIGURE 1.3 – Évolution des soumissions à l'IEEE

d'applications civiles diverses et variées reposant sur les MANETs. Nous les classons en deux groupes : les commerciales et les non-commerciales. Les premières offrent un service en ligne que le client doit payer pour en bénéficier, à l'inverse des applications du deuxième groupe. Il n'empêche que toutes ces applications présentent un fort potentiel pour les équipementiers qui construiront les terminaux permettant de les mettre en œuvre.

| | Applications | Descriptions |
|---|---|---|
| Non commerciales | Secours | Recherche de victimes et reprise après désastre, etc. |
| | Réseaux domestique ou d'entreprise | WLAN; partage d'applications, d'imprimante, etc. |
| | Education | Classe virtuelle, salle de conférence, etc. |
| | Loisirs | Jeux en réseau, robots, accès extérieur au réseau, etc. |
| Commerciales | E-Commerce | Payement électronique de n'importe quelle position |
| | Véhiculaires | • Transmission du journal, des conditions du trafic, du climat<br>• Guidage contre les accidents |
| | Affaires | • Accès dynamique aux fichiers des clients stockés dans un système central<br>• Base de données uniques pour tous les agents<br>• Bureau mobile |
| | Localisation de services | • Transfert d'appels, transfert de l'espace de travail actuel au lieu courant, etc.<br>• Publicité/Renseignement sur les services à proximité du lieu courant (station de service, imprimante, serveur, etc.) |

TABLEAU 1.2 – Applications civiles

### 1.3.3 Civiles non commerciales

#### 1.3.3.1 Secours

Mise à part les applications militaires que nous avons traitées précédemment (cf. section 1.3.1), les opérations de secours en cas d'urgences auront recourt aux MANETs pour déployer rapidement un réseau sur un terrain où les infrastructures sont inexistantes [33] ou endommagées par suite à une catastrophe naturelle, un incendie ou une attaque par un ennemi. Des études sont encours de réalisation pour intégrer les réseaux ad hoc mobiles dans l'architecture des réseaux de secours.

Aux USA, le NCS (*National Communication Systems*) a lancé le projet NS/EP (*National Security and Emergency Preparedness*) Priority Telecommunications dans le but d'étudier les exigences de conception des MANETS de secours et de reprise après désastre. Ceux-ci doivent permettre de mener des opérations efficaces et d'augmenter les chances de survie des victimes[28] suite par exemple à un bombardement, comme celui du bâtiment fédéral d'Oklahoma City, ou au passage d'un ouragan rendant le téléphone cellulaire indisponibles [34]. Le projet comprend trois composantes principales mettant l'accent sur (1) l'économie d'énergie, (2) l'utilisation efficace et appropriée de la bande passante qui peut être particulièrement limitée dans les situations d'urgence, et (3) le caractère fiable et robuste des communications.

En Europe, le programme E-SPONDER a été lancé en juillet 2010 avec la participation du CEREN (*Centre d'Études et de Recherches Économiques sur l'Énergie*). Son objectif est de développer un système global d'aide à la gestion de crise sur 3 niveaux : le centre de commandement permanent, le poste de commandement mobile et l'unité de premiers intervenants. Cette dernière déploiera un réseau MANET [29].

#### 1.3.3.2 Autres

D'autre part, les MANETs pourront servir à mettre en place un réseau local sans-fil à la maison ou en entreprise afin de pourvoir partager des machines comme une imprimante ou un disque de sauvegarde, partager des fichiers comme des fichiers multimédias ou des dossiers de clients, partager des applications coûteuses dont la mise en place est fastidieuse, etc. Un accès à Internet peut aussi être étendu ainsi pour pouvoir travailler à l'extérieur dans le jardin, au bord de la piscine ou sur une terrasse.

Dans le domaine de l'éducation, les MANETs offre aussi une perspective. Des classes virtuelles peuvent en effet avoir lieu par la mise en réseau de plusieurs salles. Un travail collaboratif sur un ensemble de données réparties sur plusieurs terminaux du MANET et se synchronisant avec une base de données distante sera réalisable. Les conférences bénéficieront aussi d'une flexibilité de mise en réseau d'appareils de projection, de prise de vidéo ou de photos, de pointage, de partage de fichiers, etc.

Enfin, dans le domaine des loisirs, les jeux en réseau gagneront à utiliser les MANETs [35] en profitant de leur flexibilité : jouer n'importe où comme dans un

33. Par exemple une zone rurale reculée ou une zone montagneuse difficilement accessible comme le cirque de Mafate à l'île de la Réunion.

34. Par exemple après le passage de l'ouragon Katrina.

35. Sony commercialise depuis 2004 la *PlayStation Portable* ou PSP, une console de jeux

parc et n'importe quand comme pendant les moments d'attente (cf. le tableau 1.2).

### 1.3.4 Civiles commerciales

Les applications commerciales reposant sur les MANETs seront essentiellement conçues dans le cadre du rôle que joueront ceux-ci en matière d'extension d'autres réseaux filaires ou sans-fil. Aussi, le RFC 2501 [5] décrit les MANETs en ces termes (*stub networks*) [36]. Cela aura tout son sens dans le cadre des réseaux de future génération (*Next Generation Network* ou NGN).

Il est en effet prévu que les MANETs fassent partie du paysage de la future génération de réseaux sans-fil, plus connue sous le nom commercial de 4ème génération [37] ou 4G. Au sein de cette génération, l'effort est mis sur l'intégration transparente de différents types de réseaux sans-fil à large bande avec le réseau dorsal filaire d'Internet. Ainsi les différents flux de voix [38], de multimédia [39], et de données se trouveront encapsulés dans des paquets IP au cœur du réseau. On parle alors de convergence des flux. Cette convergence se basera sur une architecture standardisée appelée IMS (*IP Multimedia Subsystem*) [40].

La figure 1.4 illustre, par les chevauchements des réseaux sans-fil (réseaux satellitaires, cellulaires [41], WPAN, WLAN et MANET) entre eux, d'une part, et avec le cœur du réseau, d'autre part, les différentes intégrations et interopérabilités entre réseaux à prendre en considération dans les NGN. Cela permettra à l'utilisateur d'accéder à un panel de services à tout moment, n'importe où et depuis n'importe quel terminal. Ces services seront des adaptations de services existants aujourd'hui ou des services futurs répondants mieux aux besoins de mobilité des utilisateurs[32].

Dans le paysage 4G, les MANETs amèneront ces services dans les zones non desservies par les réseaux à infrastructure à cause d'une mise en œuvre difficile et/ou coûteuse [42]. Ils assureront une plus grande couverture de réseau à moindre

---

vidéo ayant la capacité de se connecter en réseau ad hoc à seize machine au maximum grâce à la technlogie WiFi. Cette PSP a subit des évolutions depuis, et en 2009, selon le site GAMEKULT, Sony a annoncé qu'elle « s'est écoulée à 52,9 millions d'unités dans le monde, dont 17 millions sur le Vieux Continent »[30].

36. Le trafic traversant ce type de réseau doit avoir sa source ou sa destination dans le réseau même mais jamais il ne doit y transiter seulement. La cause en est les limites physiques des MANET à savoir celles des ressources énergétiques et de la bande passante.

37. Cette génération est aussi désignée par le nom IMT évoluées (*International Mobile Telecommunications-Advanced* ou *IMT-Advanced)*.

38. Avant l'arrivée de technologies de voix sur IP (VoIP) tels que les protocoles MEGA-COP, MGCP, SIP et H.323 toutes les communications téléphoniques traversaient au départ le réseau téléphonique commuté (RTC), ensuite, avec l'avènement du GSM, certaines traversaient plutôt les réseaux mobiles.

39. Autrefois le flux multimédia était diffusé seulement par les antennes de télévision ou par les satellites, ensuite aussi par le *câble* à travers des offres commerciales telles que celles de *Canal+*. Depuis quelques années, des entreprises comme SFR et Orange offrent l'accès à ce flux par internet à travers des offres tels que le *triple play*.

40. Cette convergence abaissera les coûts de construction des réseaux en mettant en œuvre des WLAN et des WPAN dont l'avantage est de ne pas nécessiter pas de licences.

41. Le détail des exigences de performance des réseaux cellulaires de 4ème génération, a été établit en 2008 par la section UIT-R (*radio-commnication* de l'*Union Internationale des Télécommunications*), . Ayant en leur cœur un réseau IP, ces réseaux cellulaires offriront un débit de l'ordre de 1Gbps en haute performance et de 100 Mbps en basse performance [31].

42. Au cours du concours de dissertations par les étudiants organisé par Symbian en 2008, Milen Nikolov avait soumis un essai décrivant un scénario d'utilisation des MANETs dans

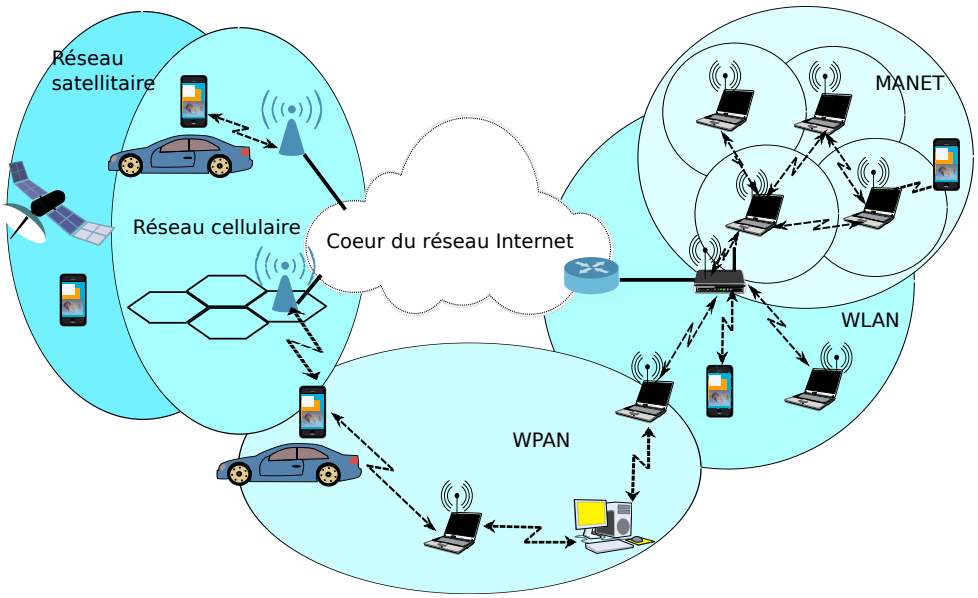


FIGURE 1.4 – Réseaux 4G [1]

coûts et permettront d'augmenter ainsi le nombre de clients en conquérant des nouveaux. Des nœuds connectés à Internet joueront pour cela le rôle de point d'accès au cœur du réseau. Mais il doit être prévu que ces services continuent à fonctionner même en l'absence de connexion à Internet. [34] décrit une manière d'assurer l'interopérabilité entre un MANET et un réseau à infrastructure grâce à la technologie NEMO (*Network Mobility*) [43].

Plusieurs projets visant cette intégration ont été lancés. Par exemple le projet DAIDALOS (*Designing Advanced network Interfaces for the Delivery and Administration of Location independent, Optimised personal Services*), lancé en 2004 et arrêté en 2008, a fait partie du thème IST (*Information Society Technologies*) du programme européen FP6 (*Framework Program Six*) [35]. On peut aussi noté le projet finlandais SESSI (*Seamless Service Interworking in Heterogeneous Mobile and Ad Hoc Networks*) [36] lancé en 2004 et le projet français SARAH (*Service Avancés pour Réseaux Ad Hoc*) soutenu par l'Agence Nationale de la Recherche (ANR) lancé en 2007 et achevé en 2010.

Parmi ces services on peut citer le E-Commerce, l'accès à des fichiers stockés sur sur un système centralisé, le transfert d'appels ou encore les renseignements concernant des services à proximité d'un lieu donné. Certains d'autres comme l'accès par des agents commerciaux à une base centrale commune de données seront plus efficaces et renforcés. De nouveaux services verront aussi le jour comme le guidage pendant la conduite automobile afin de prévenir les accidents ou le transfert de bureau de travail au fur et à mesure des déplacements d'un

le métro de New York où la couverture du réseau n'est pas suffisante alors qu'un million et demi de personnes par an emprunte ce métro. Selon cet essai, tous les jours les entreprises de téléphonie mobile perdent 5 042 263 clients potentiels. Organiser les smartphones des passagers en MANETs pourrait être une solution [33].

43. Il s'agit d'un protocole de routage qui opère entre un réseau mobile, comme un WLAN installé dans un train, et un réseau fixe à infrastructure.

agent (cf. le tableau 1.2).

Le commerce des services en réseau MANET nécessitera le contrôle de l'accès à ceux-ci et l'assurance de la sécurité des opérations de vente et en particulier leur payement. Dans les réseaux à infrastructure, cela est le rôle des infrastructures AAA. C'est ce que nous tenterons de mettre en œuvre dans ce travail. Pour cela, des défis liés à la nature même de ces réseaux doivent être relevés

## 1.4 Défis

Les solutions de sécurité qui ont été conçues pour les réseaux filaires ne sont pas nécessairement les mieux adaptées au contexte sans-fil et mobile en général. Cela est d'autant plus vrai quand il s'agit des réseaux mobile ad hoc que nous étudions et qui présentent des problèmes à des niveaux divers de la pile de protocoles notamment physique, liaison de données et routage.

### 1.4.1 Liens sans-fil

Réaliser de la sécurité dans un contexte sans-fil est difficile pour des raisons liées à la nature des ondes électromagnétiques. En effet, leur propagation est soumise à des aléas tels que les réflexions, les interférences, le fading [44] qui provoquent des erreurs de transmissions et rendent les connexions sporadiques. La conception des couches basses de la pile de protocoles prend donc en compte ces phénomènes et sont plus complexes et plus perfectionnée que celles des réseaux filaires. Les perfectionnements se trouvent, en particulier, intégrés dans la couche liaison de données [45] dont les algorithmes ont été conçus pour diminuer le taux d'erreurs. Toutefois, ces mécanismes engendrent une diminution de la bande passante disponible par rapport à une technologie filaire semblable.

De plus, l'atténuation du signal électromagnétique implique une portée limité réduisant le nombre de connections possibles pour un nœud donné. Les conditions de connexions se trouvent ainsi restreintes. Mais ces problèmes sont en partie résolus grâce au routage multi-sauts dans les MANET contrairement au réseau locaux construits autour d'un point d'accès. En contre partie, le routage implique une signalisation fastidieuse provoquant aussi une diminution de la bande passante. Enfin, les ondes électromagnétiques sont soumises à l'effet *Doppler-Fizeau* quand les nœuds se déplacent. Pour les nœuds ne se déplaçant pas trop vite, un simple dimensionnement adéquat de la largeur des canaux est suffisant.

Un autre aspect non moins important à noter est le fait que les ondes électromagnétiques ne sont pas confinées. L'écoute des transmissions et la transmission même de trafic ne nécessitent pas un accès matériel comme cela est le cas dans les réseaux filaires. La sécurité vis-à-vis des tiers est par conséquent difficile à assurer. Une des premières mesures qui ont été prises est le saut de fréquences utilisé dans la technologie de transmission FHSS (*Frequency-Hopping Spread Spectrum*). D'autre mesures ont recours à la cryptographie pour sécuriser l'accès au canal comme celle utilisant WEP, déjà dépassé, et plus tard WPA (cf.

44. Il s'agit d'un phénomène particulier d'atténuation régulière à fréquence assez basse, de l'ordre de quelques fractions de Hertz, et dû à des battements entre les signaux transmis par une onde directe et ceux transmis par une onde ayant subie une ou plusieurs réflexions.

45. La norme IEEE 802.11 prévoit l'acquittement de chaque message reçu.

chapitre 1, section 2.6).

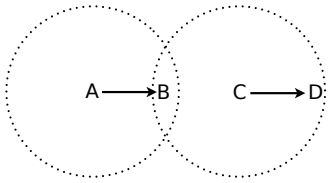


FIGURE 1.5 – Phénomène de la station cachée

Par ailleurs, toujours en raison du non confinement, le signal issu d'un nœud atteint tous les nœuds à sa portée rendant plus difficile l'installation d'un canal sans-fil et les risques de collisions des messages émis ou reçus par des nœuds voisins importants. Un des problèmes connus à ce propos est le problème de la station cachée présenté à la figure 1.5 : le nœud A est en train d'envoyer un message à B quand C, incapable de capter les messages de A, transmet à son tour un message à D. B, se trouvant à la fois dans les champs de transmission de A et de C, ne réussit pas à recevoir le message de A car celui-ci rentre en collision avec le message dans la conception de C. La prise en compte de ce phénomène entraîne une plus grande complexité de la couche de liaison de données. Ainsi la norme 802.11, par exemple, permet au nœud A de réserver le canal en envoyant un message RTS (*Request To Send*) à B qu'il acquitte par un message CTS (*Clear To Send*) reçu par C qui note dorénavant qu'il ne lui est pas possible d'émettre pour un certain temps.

CSMA/CA (*Carrier Sense Multiple Access/Collision Avoidance*), un protocole incluant ce mécanisme, fait partie de la fonction distribuée DCF (*Distributed Coordination Function*) de la norme IEEE 802.11. DCF a été conçu pour assurer un accès distribué au canal contrairment à la fonction PCF (*Point Coordination Function*).

### 1.4.2 Réseau multi-sauts et nœuds mobiles

Le routage multi-sauts est une opération compliquée dans les MANETs à cause de la mobilité des nœuds qui assurent eux mêmes cette tâche. Des changements fréquents de topologie provoquent une charge du réseau supplémentaire et nécessaire pour trouver de nouvelles routes, de nouveaux partitionnement. Une forte mobilité peut provoquer, de plus, la perte de paquets de routage. L'usage de protocoles de routage plus complexes consommant une part plus grande de la bande passante devient alors inévitable. Le protocole AODV notamment peut entraîner jusqu'à 80% de charge du réseau [37]

### 1.4.3 Absence d'une administration centrale

Une administration centralisée a pour fonctions de distribuer les clés de sécurité, de contrôler les droits et les accès (cf. Chapitre 1) de surveiller les agissements des utilisateurs, etc. Une telle administration n'existe pas forcément dans les réseaux ad hoc mobiles.

Dans les applications militaires et de secours, tous les nœuds du réseau appartiennent à une même autorité, par exemple une unité militaire ou une équipe de secours et ont un but commun.

Cependant, les technologies sans fil se sont sensiblement améliorées depuis les années 1990 et des dispositifs peu coûteux basés sur la norme 802.11 ont envahi le marché. Le déploiement des MANETs pour des applications publiques est ainsi devenu réalisable. Des exemples incluent des applications commerciales et non commerciales (voir tableau 1.2).

Dans ces réseaux, les nœuds n'appartiennent pas nécessairement à une même organisation ni à une même autorité et ils ne poursuivent peut-être même pas un but commun. En outre, les MANETs publics pourraient être plus grands et avoir une durée de vie plus longue. De plus ils peuvent être complètement autonomes, signifiant que le réseau fonctionnerait grâce à l'opération des seuls utilisateurs.

#### 1.4.3.1 Absence de relations de confiance

Les nœuds n'ont pas forcément une même autorité. Parfois même, celle-ci est absente ou inexistante. Une relation de confiance à priori entre les nœuds n'est donc pas forcément disponible. Elle doit alors être établie par des mécanismes spécifiques conçus en fonction du scénario et de l'application offerts.

### 1.4.4 Contexte distribué

Les algorithmes de gestion, qu'il s'agisse d'algorithmes de fonctionnement des couches basses du réseau ou de mécanismes de gestion des clés cryptographiques [38] deviennent des algorithmes coopératifs dont l'écriture est donc plus complexe, la validité plus délicate à établir, et l'attaque plus facile à réaliser en se concentrant sur les maillons faibles [46]. Les attaques contre les mécanismes de sécurité eux mêmes deviennent plus faciles [38].

### 1.4.5 Nœuds égoïstes

Dans ce contexte distribué et sans administration centrale, il est très simple pour un nœud de manipuler un protocole d'acheminement afin d'économiser les ressources (énergie, puissance de calculs, etc) d'une façon *égoïste*. Le besoin de coopération entre les nœuds pour assurer le fonctionnement du réseau est en conflit avec l'intérêt individuel de chaque nœud visant à ne dépenser de l'énergie (une ressource précieuse, car dans la majorité des cas les dispositifs mobiles sont alimentés par batterie ayant une durée de vie limitée) que pour les flux de trafic qui leur sont destinés ou pour lesquels ils sont origines [39].

### 1.4.6 Nœuds malveillants

Un certain nombre de problèmes de sécurité qui existent déjà dans les réseaux filaires prennent dans les MANET une dimension toute particulière notamment

46. La déviation du comportement des nœuds par rapport à l'exécution définie par un protocole distribué a été déjà constatée dans les réseaux pair-à-pair. Des modifications locales de protocoles sont faciles à réaliser afin d'exploiter le système sans pour autant y contribuer activement.

ceux liés au fonctionnement des mécanismes de base du réseau comme le routage. En diffusant, des nœuds malveillants peuvent envoyer de fausses informations de routage ou encore simplement les modifier.

### 1.4.7 Contrainte d'énergie

L'énergie limitée des nœuds amène à choisir des protocoles et des opérations de réseau peu consommatrices d'énergie. Cela concerne aussi bien les opérations des couches physique, liaison de données et routage que les mécanismes de sécurité.

### 1.4.8 Terminaux hétérogènes

L' hétérogénéité des terminaux aggrave les difficultés ci-dessus. Leurs caractéristiques peuvent effectivement être dissemblables telles que celles se rattachant aux technologies sans fil, aux versions des différentes normes suivies, aux capacités des batteries, aux puissances de calculs des processeurs, aux types du système d'exploitation et des programmes, etc.

Enfin, le facteur d'insécurité lié au vol des terminaux : les utilisateurs les emploient n'importe où contrairement par exemple à des ordinateurs fixes qui restent enfermés dans des pièces.

## 1.5 Conclusion

La flexibilité des MANETs a un coût. Mettre en œuvre des applications à usage public n'est pas facile dans un réseau sans-fil, multi-sauts, distribué, sans administration centrale définie, où les terminaux sont hétérogènes et où le comportement des nœuds peut être égoïste voire malveillant. La présence de tels utilisateurs dans les réseaux n'est pas nouvelle. Nous l'avons montré au premier chapitre. Néanmoins, leurs actions sont plus prégnantes à cause des caractéristiques des MANETs.

Il s'agit donc d'apporter des solutions de sécurité pour les services reposant sur les MANETs. Une solution globale s'occuperait d'assurer les éléments de base de la sécurité à savoir l'authentification des nœuds, la confidentialité et l'intégrité de leur flux, l'absence de rejeux et la non-répudiation. De tels éléments sont pris en charge par des architectures AAA.

# Chapitre 2

# Sécurité des réseaux à infrastructure

## Sommaire


**2.1 Introduction** . . . . . **22**
**2.2 Vocabulaire et méthodologie** . . . . . **22**
2.2.1 Définition de la sécurité . . . . . 22
2.2.2 Méthodologie . . . . . 22
2.2.3 Que veut-on sécuriser ? . . . . . 24
2.2.4 Contre quels dangers ? . . . . . 26
2.2.5 Comment ? . . . . . 27
**2.3 Techniques de contrôle d'accès** . . . . . **28**
2.3.1 Histoire . . . . . 28
2.3.2 Droits d'accès et domaine . . . . . 30
2.3.3 Identification contre Authentification . . . . . 30
2.3.4 Authentification par mot de passe statique . . . . . 31
2.3.5 Authentification par mot de passe dynamique (OTP) 32
2.3.6 Garde-barrières . . . . . 32
**2.4 Outils cryptographiques et services de sécurité** . **34**
2.4.1 Algorithmes de chiffrement . . . . . 35
2.4.2 Hachage cryptographique . . . . . 38
2.4.3 Services de sécurité . . . . . 39
2.4.3.1 Authentification cryptographique . . . . . 39
2.4.3.2 Confidentialité des données . . . . . 39
2.4.3.3 Intégrité des données . . . . . 40
2.4.3.4 Non répudiation . . . . . 40
2.4.3.5 Absence de rejeu . . . . . 40
**2.5 Relation de confiance et gestion de clés** . . . . . **40**
2.5.1 Hiérarchisation de la confiance . . . . . 41
2.5.2 Renforcement de la vigilance en environnement distribué . . . . . 42
2.5.3 Partage de clés . . . . . 43
**2.6 Protocoles de sécurité** . . . . . **44**
**2.7 Conclusion** . . . . . **45**

## 2.1 Introduction

La problématique de la sécurité dans les réseaux en général est étudiée depuis longtemps. Pourtant, en raison de la confiance que se témoignaient les universités et organismes qui fondèrent DARPANET, en raison aussi de la focalisation des chercheurs sur les problèmes d'échange d'information, il est à remarquer que le soucis de la sécurité telle que nous la concevons aujourd'hui ne faisait pas, à ses débuts, partie intégrante de la conception d'Internet. Ce n'est qu'à dater de la massification de l'accès à ce réseau et avec l'augmentation d'un trafic de plus en plus riche en informations exploitables par des tiers plus ou moins malveillants que la sécurité a été progressivement intégrée à l'ensemble des spécifications d'Internet.

## 2.2 Vocabulaire et méthodologie

### 2.2.1 Définition de la sécurité

Selon Littré [40] la sécurité est une « tranquillité d'esprit bien ou mal fondée dans une occasion où il pourrait y avoir sujet de craindre. Sécurité se prend aussi pour indiquer non pas la tranquillité d'un seul homme, mais celle d'un peuple, d'une association, d'une corporation entière ».

Selon Larousse [41] la sécurité est une « situation dans laquelle quelqu'un, quelque chose n'est exposé à aucun danger, à aucun risque d'agression physique, d'accident, de vol, de détérioration ».

Selon le TLFi [3], la sécurité est une« situation objective, reposant sur des conditions matérielles, économiques, politiques, qui entraîne l'absence de dangers pour les personnes ou de menaces pour les biens et qui détermine la confiance ».

Enfin, selon Robert [42] la sécurité est un « état d'esprit confiant et tranquille d'une personne qui se croit à l'abri du danger. Situation, état tranquille qui résulte de l'absence réelle de danger (d'ordre matériel ou moral). Organisation, conditions matérielles, économiques, politiques, propres à créer un tel état ».

On voit que certains auteurs s'attachent d'avantage au ressenti, qu'il soit justifié ou non alors que les auteurs plus récents s'attachent d'avantage à la situation objective. S'agissant de systèmes informatisés et plus particulièrement de réseau, nous définirons la sécurité comme un état ressenti par la communauté des utilisateurs du système tout en retenant qu'il s'agit comme le note Robert de l'ensemble des conditions objectives à mettre en place pour créer un tel état.

### 2.2.2 Méthodologie

Au vu de la définition ci-dessus, on se rend compte que les problèmes de sécurité [1] sont de nature globale et appellent donc des solutions de nature globale. En telle matière, il est important de savoir identifier les risques encourus

1. Sur toutes ces questions de sécurité, il a été produit un volume important de travaux. Sans pouvoir les citer tous, on peut trouver une référence opérationnelle récente avec le Référentiel Général de Sécurité (RGS), résultant en particulier des travaux de l'Agence Nationale de la Sécurité des Systèmes d'Information (ANNSSI) et de la Direction Générale de la Modernisation de l'Etat (DGME), cf. arrêté du Premier Ministre du 6 mai 2010 (JO du 18 mai 2010) `http://www.references.modernisation.gouv.fr/sites/default/files/RGSv1.0.pdf`

et leurs origines, de mesurer les coûts associés aux ruptures de sécurité et de les mettre en rapport avec les coûts directs ou indirects des mesures mises en place, de ne pas oublier enfin, qu'avec un temps suffisamment long [2] et des moyens suffisants, il est possible de venir à bout de bien des barrières.

Cependant, l'art de l'ingénieur et les connaissances des chercheurs permettent d'étudier ce que l'on pourrait appeler la sécurité partielle : des éléments de construction facilitant la mise en place de solutions globales. Ceci amène à se poser trois questions : que veut-on sécuriser ? contre quels dangers ? et comment ?

Nous poserons ces questions à propos des réseaux d'ordinateurs, que nous désignons aussi par réseaux informatiques ou tout simplement *réseau*. Au premier titre de ces réseaux figure naturellement Internet. Notons bien que les meilleurs mécanismes de protection de ces machines sont comme toujours inutiles face aux négligences humaines (clefs oubliées sur une porte, alarmes débranchées, mots de passe trop simples, écrits dans un carnet bien en évidence à côté de l'ordinateur ou sur une étiquette collée sous le clavier, réponses inconsidérées à des questions du système d'exploitation, etc) qui dénotent un manque de formation des personnes aux impératifs de la sécurité.

Si l'on considère un ensemble d'ordinateurs, la sécurité des données qu'ils renferment résulte d'une organisation de la sécurité, appuyée sur la qualité des mécanismes de protection relevant des systèmes d'exploitation de ces ordinateurs. Lorsque ces différentes machines interconnectées *via* un réseau, nous dirons que *le réseau est sûr* si l'ensemble de ces machines désormais interconnectées n'a pas plus de problèmes de sécurité qu'il n'en avait alors que les machines étaient séparées. En d'autres termes, et cela sera notre définition, un réseau sûr est un réseau qui n'introduit pas de problèmes de sécurité supplémentaires. Par exemple, si l'une des machines tombe en panne matérielle cela ne doit avoir aucune répercussion sur le fonctionnement des autres machines, par exemple encore, le fait qu'une machine soit infectée par un virus informatique ne doit pas entraîner *ipso-facto* la transmission de ce virus à d'autres machines de l'ensemble.

Notons que les principaux systèmes d'exploitation conçus pour les ordinateurs de taille moyenne ou grande dans les années 1970-1980 (Unix et toutes ses variantes, VMS, VM/CMS) intégraient complètement dès leur conception la notion de sécurité interne mais pas celle de sécurité du réseau. Il a fallu attendre les années 1990 pour que des systèmes de conceptions plus récentes tels que Plan9 [3] ou Inferno [4] intègrent la dimension réseau et sécurité implantée au cœur même des systèmes d'exploitation et ce dès leur conception. Au contraire, les premiers systèmes d'exploitation pour micro-ordinateurs ne traitaient pas du tout de la sécurité. La mise en place de celle-ci a toujours été délicate et source de nombreuses faiblesses et de mises à jour fréquentes sans que l'on puisse pour autant savoir quel niveau de sécurité réel est atteint [5], ni même jusqu'à quel

2. Si pour un être humain quelques minutes peut sembler un temps très bref, pour un microprocesseur quelques dixièmes de secondes permettent d'exécuter de l'ordre d'un milliard d'instructions...

3. On pourra par exemple se reporter à la conférence *Plan 9 from Bell Labs* 4ème édition cf. `http://plan9.bell-labs.com/plan9/index.html`

4. Voir par exemple le site de la société *vita nuova* qui distribue ce système d'exploitation héritier direct de Plan9 cf. `http://www/vitanuova.com/index.html`

5. La multiplication des systèmes anti-virus, anti-adware, des mises à jour de sécurité est sur ce point une indication précieuse

point la connexion de telles machines à un réseau ne constitue pas en elle même un danger.

### 2.2.3 Que veut-on sécuriser ?

Les réseaux sont constitués de divers équipements d'une part et de liens filaires (cuivre, fibre optique) ou non filaires (infrarouge, onde radio en champ libre, faisceau hertzien dirigé) qui les relient d'autre part. Toute ou partie de ces équipements peuvent être gérés, conduits, animés, par des programmes adaptés et plusieurs sortes de données y sont stockées. Certaines d'entre elles peuvent être l'objet de transferts selon des protocoles appelés *protocoles de réseaux*. Dans ce cadre, la sécurité concerne celle du matériel, celle des programmes, celle des données, et celle des protocoles. Nous verrons qu'elle s'étend au delà.

**Matériel** Mis à part les ordinateurs que les réseaux relient, le matériel inclut aussi, lorsqu'ils sont présents, les équipements spécialisés : répéteurs, autocommutateurs privés (PABX) ou publics, répartiteurs (hubs), commutateurs (Swich), routeurs, garde-barrières (firewalls), modems, serveurs, etc. Certains équipements pouvant d'ailleurs regrouper au sein d'un même boîtier plusieurs des fonctionnalités du matériel ci-dessus. La limitation de l'accès à ces matériels (enfouissement, salles spécialisées, équipes de gardiennage et de sécurité), participe à la sécurité de l'ensemble.

**Programmes** Les programmes comportent naturellement les systèmes d'exploitation – y compris les pilotes de périphériques – ainsi que les logiciels, programmés ou microcodés, gérant les différents mécanismes de réseaux. Ils comportent aussi des programmes liés à certains protocoles selon le modèle client/serveur et l'on parle alors de *services*, par exemple telnet/telnetd pour permettre un service de connexions à distance, FireFox/Apache pour consulter un site web que l'on a établi, FileZilla/ftpd pour télécharger des fichiers installés dans un serveur, ChatZilla/ircd pour communiquer avec la messagerie instantanée sur Internet, Mozilla Sunbird/serveur CalDAV fournissant un service d'édition et de consultation des calendriers, etc. Des associations plus complexes peuvent être rencontrées autour de plusieurs protocoles comme ThunderBird/popd/smtpd. On trouve aussi des logiciels de type pair-à-pair tels que Skype ou BitTorrent.

D'autre services fonctionnant en mode client/serveur ne ciblent pas directement les utilisateurs mais le réseau qu'ils utilisent en permettant une meilleure gestion à distance et plus d'autonomie. On parle alors de *services réseau*. Parmi ces services on peut citer DHCP (Dynamic Host Configuration Protocol) qui automatise l'attribution aux machines d'adresses IP, DNS (Domain Name Server) qui facilite la recherche du domaine auquel appartient une page web , Radius et Kerberos qui centralisent le contrôle d'accès au réseau, NTP (Network Time Protocol) qui permet de synchroniser les machines, SNMP (Simple Network Management Protocol) qui permet aux administrateurs du réseau de gérer et superviser les équipements du réseau à distance, etc.

**Données** Les données sont essentiellement de deux sortes : celles qui servent aux fonctionnement du réseau comme les tables de routage, les bases de données

des clients, les fichiers relatifs aux droits d'accès, etc. et celles qui ne relèvent pas du fonctionnement du réseau comme les documents et les archives, les photos, les vidéo, etc.

Une telle énumération ne saurait être exhaustive. La variété des réseaux construits ou imaginés à ce jour est suffisamment grande pour garantir que l'on puisse trouver d'autres exemples. Mais, dans chaque catégorie, l'énumération présentée donne une bonne idée de ce que l'on peut raisonnablement rencontrer. Par ailleurs, les évolutions technologiques : augmentation de la capacité des mémoires comme celle de la puissance des processeurs qui résultent de la loi de Moore se combinent avec des possibilités de miniaturisation toujours plus grandes pour autoriser la conception et la fabrication d'objets extrêmement complexes. Il devient alors difficile de discerner la frontière de ce qui relève du matériel spécialisé dans la gestion du réseau avec ce qui relève simplement du micro-ordinateur de bureau.

**Protocoles** Selon le TLFi, étymologiquement, un protocole, du grec $\pi\rho\omega\tau o$ « premier » et $\kappa o\lambda\lambda\alpha$ « gomme (arabique) » est utilisé au XVII$^{\text{ème}}$ siècle pour désigner « un formulaire contenant la manière dont les grands princes traitent dans leurs lettres ceux à qui ils écrivent » et au XIX$^{\text{ème}}$ siècle le « Protocole *diplomatique* : c'est la règle du cérémonial à suivre dans les rapports politiques officiels entre les États aussi bien qu'entre les ministres ».

Ce mot est ensuite entré, dans les années 1960, dans le vocabulaire de l'informatique pour désigner un « ensemble de conventions qui permettent à deux matériels de se connecter entre eux » on parle alors souvent de *shake hand* ou poignée de main, les normes RS 232C, IEEE 488 ou les standards bus S100, bus PC/AT donnent de bons exemples de tels protocoles[6]. Dans ces cas, les échanges conventionnels sont des échanges de signaux logiques TTL (*Transistor-Transistor-Logic*). Dans les années 1970, cette notion a progressivement évoluée pour désigner un ensemble de règles permettant à deux entités de se connecter entre elles, que ce soit directement ou à travers un réseau. Un protocole implique au moins deux entités. Sa conception est tout aussi importante que son implémentation. Aucune des deux ne doit comporter de défauts pouvant devenir plus tard sources de dangers pour les matériels ou les programmes.

Un protocole se déroule en un ou plusieurs échanges qui peuvent prendre la forme de messages. Un message est formé d'un en-tête et d'un corps ou contenu pouvant être vide ou rempli de données de diverses sortes comme présenté plus haut.

Pour simplifier la conception de programme de traitement et suivant les idées de David Parnas [43], on procède par abstractions successives. Ainsi chaque programme s'exécute sans qu'il lui soit nécessaire de connaître les détails des traitements qu'il demande et qui sont pris en charge par des programmes de niveau inférieur. Un message créé par un programme sur une machine donnée peut ainsi être remis à un autre programme sur cette même machine pour qu'il le traite. Des traitements en chaîne par des programmes différents sur la même machine

6. Si les bus S100 relèvent aujourd'hui davantage de la muséographie, le bus RS232 avec son connecteur DB9 est encore présent sur beaucoup de micro-ordinateurs, le bus IEEE 488 reste très présent dans l'industrie en particulier en instrumentation et en contrôle de processus et le bus PC/AT après diverses évolutions techniques reste le standard des machines dites *compatibles*.

peuvent avoir lieu jusqu'à l'arrivée du message à un dernier programme qui l'envoie, à travers un canal de communication, à un programme homologue distant. On dit qu'une telle architecture est une architecture *en couche*. Un programme implémentant un niveau d'abstraction s'appelle une *couche de réseau*. Les liens entre les différentes couches sont souvent représentés par une pile. Comme à chaque niveau d'abstraction correspond un ou plusieurs protocoles, on appelle cette pile la *pile de protocoles*.

On connaît deux modèles de piles de protocoles : le modèle OSI à sept couches qui est le plus abouti intellectuellement et le modèle TCP/IP à cinq couches qui reste pourtant le plus utilisé pour des raisons d'antériorité matérielle. Prenons une machine émettrice d'un message, dans l'ordre, les programmes exécutés sont ceux de la couche application, ensuite transport, puis réseau, ensuite liaison de données et enfin physique. Les programmes de la couche applicative sont nombreux et divers contrairement à ceux des autres couches. Plus une machine est puissante en capacité de calculs et de stockage, plus elle contiendra d'applications et plus elle pourra en faire tourner plusieurs au même temps.

**Intégrité et continuité de service** Comme toute autre entité organisée, un réseau a pour tâche première de se maintenir en état de rendre les services pour lesquels il a été conçu. Cette fonction que l'on appelle souvent *maintien d'un fonctionnement normal* apparaît *in fine* la pierre de touche d'une sécurité maîtrisée. Les opérateurs d'un réseau, que l'on désigne par là un petit groupe d'amis réunissant leurs machines pour jouer en groupe ou les responsables d'une grande société internationale ou nationale désirent assurer le fonctionnement global du réseau qu'ils gèrent. Il faut pour cela un certain nombre de garanties de fonctionnement de la totalité, ou, en cas de redondance, d'une partie suffisante des équipements qui le constituent. Pourtant, cette garantie, nécessaire, ne suffit pas à assurer la continuité d'un fonctionnement fiable de l'ensemble du réseau.

### 2.2.4 Contre quels dangers ?

Qu'il s'agisse de la rupture d'un câble provoquée par une fausse manœuvre d'un engin de travaux publics ou du brouillage d'une communication hertzienne par le fonctionnement d'un four à micro-ondes, le fonctionnement d'un réseau est tributaire d'événements extérieurs parfois imprévisibles. On parlera alors de *circonstance*. Il existe aussi des dangers provoqués par des personnes utilisant le réseau c.-à-d. des *acteurs*. Il est par ailleurs des dangers provenant d'autres machines. Lorsque la distinction entre machine et personne n'a pas d'utilité, nous désignerons indifféremment les uns et les autres sous le nom d'**entités**.

Nous avons identifié deux types d'acteurs pouvant constituer une menace pour un réseau : les malintentionnés et les maladroits[7]. Les premiers, appelés aussi *adversaires* ou *attaquant*, agissent délibérément et sont à l'origine de plusieurs types d'actions visant directement ou indirectement à altérer voire détruire le bon fonctionnement du réseau. On appelle ces actions des *attaques*. Les seconds méconnaissent en général les problèmes de sécurité, par naïveté ou faute de formation. Pour autant, les dangers qu'ils provoquent peuvent être aussi lourds de menaces que l'action délibérée des premiers.

7. A la différence des premiers, ceux-la hélas ne s'arrêtent jamais pour se reposer.

L'abondante littérature sur la sécurité comporte, outre les ouvrages théoriques, les avis des CERT [8] qui recensent des dizaines et des dizaines de type d'attaques, leurs noms à eux seuls ne manquent pas d'une certaine poésie, et leur nombre augmente tous les jours. Nous n'avons ni le temps ni l'espace pour une étude détaillée – d'ailleurs ce n'est pas le propos de cette thèse. On peut toutefois noter que les attaques sont de deux catégories : les attaques actives et les attaques passives. Les attaques actives peuvent aller du vol, à la détérioration, la modification, ou la destruction du matériel et/ou des logiciels et/ou des données. Les attaques passives consistent à capter les messages de certaines communications [9].

On peut grossièrement classer les problèmes de sécurité en quatre thèmes bien connus. Un adversaire cherchera en général à accéder au réseau (matériel, programmes, données) même s'il n'y a pas droit. On est alors face à un problème de *contrôle d'accès*. Il pourra aussi essayer de découvrir le contenu de certains messages comme décrit ci-dessus et l'on est alors face à un problème de sûreté, plus connu sous le nom de *confidentialité*. Il pourra par ailleurs tenter de changer le contenu de ces messages. C'est le problème d'*intégrité*. Enfin, il peut essayer de nier avoir envoyé un trafic donné et l'on est alors face à un problème de *répudiation*.

Dans cette thèse nous ne nous intéresserons pas à la résolution des problèmes de sécurité liés au matériel, ni à régler tous les problèmes de sécurité liés aux données, aux programmes et aux protocoles qui peuvent s'y rapporter. Nous n'examinerons pas les comportements recommandés aux usagers pour accroître la sécurité et les maintenir en cohérence avec les techniques de sécurité exposés. Nous examinerons plutôt la sécurité de certains protocoles permettant de demander l'accès à certaines données ou de les transférer.

### 2.2.5 Comment ?

Les solutions de sécurité que l'on trouve de nos jours sont diverses. Le plus souvent, elles sont aussi partielles, c'est à dire qu'elles n'adressent qu'une partie de la sécurité. Par ailleurs, elles sont souvent tellement complexes qu'il est difficile de les appréhender dans leur entièreté. Cela provient d'abord, de ce qu'en raison de la grande capacité en mémoire et de la grande puissance des processeurs, il y a généralement plusieurs programmes dans la même machine. Nous avons évoqué ce fait précédemment. Cela s'explique aussi par la volonté de parer plusieurs sortes d'attaques à la fois en utilisant un minimum d'équipements dédiés au réseau.

Plusieurs stratégies et techniques ont été développées pour résoudre les problèmes de sécurité. Nous présenterons celles qui pourraient protéger les données et les programmes, surtout celles qui peuvent se rapporter à des protocoles. Nous traiterons expressément dans la section suivante des techniques de contrôle de

8. Les CERT (*Computer Emergency Response Team*) ont pour mission de centraliser et de traiter les alertes et les incidents de sécurité ou attaques

9. Par exemple en réglant l'interface de réseau d'une machine, se trouvant sur le chemin entre la source et la destination du trafic espionné, en mode *promiscuous*. En fonctionnement normal, une interface de réseau ne capte que le trafic qui est destiné à la machine sur laquelle elle est installée. Mais en mode *promiscuous*, cette interface capte tout le trafic qui traverse le canal de communication. Des logiciels connus sous le nom de *sniffers* permettent alors de capturer ce trafic. Des logiciels connus sous le nom d'analyseurs de trafic intégrés ou non avec les *sniffers* en permettent une analyse précise.

l'accès aux réseaux ou plus précisément de l'accès aux machines qui s'occupent de la gestion de ce contrôle. Nous présenterons des classes de techniques spécialisées dans le cryptage des données et d'autres dans leur protection des modifications. Enfin nous montrerons que ces techniques sont basées sur la constitution de relations de confiance entre les entités qui communiquent à travers le réseau. Des techniques ont été aussi élaborées à cette fin.

Avant de laisser une entité accéder d'une façon ou d'une autre à un réseau, il importe de déterminer l'identité de cette entité, de la contrôler, et de déterminer enfin quelles sont les actions qui lui seront autorisées. C'est la phase « AA » c.-à-d. *Authentification et Autorisation*. Cette phase utilise des techniques de contrôle d'accès que nous allons maintenant examiner.

## 2.3 Techniques de contrôle d'accès

Une conception totalement paranoïaque de la sécurité voudrait qu'avant de demander chaque exécution d'un travail, quel qu'il soit, le demandeur apporte la preuve de son droit à présenter cette demande en précisant par exemple le lieu d'où il fait la demande et le nom de l'entité qui est à l'origine de cette demande. Au reste, une sécurité attentive vérifierait sans doute de plus l'opportunité de la demande [10]. Cependant malgré la sécurité (toujours faillible) d'une telle organisation, la lourdeur d'une telle procédure n'échappe à personne.

C'est pourquoi on préfère définir une zone de confiance délimitée dans le temps et dans l'espace qui autorise une entité de cette zone à effectuer un certain type d'action. En ce qui concerne la délimitation dans l'espace, on parle souvent de *domaine*, alors qu'en ce qui concerne la délimitation dans le temps on parle en général de *session*.

### 2.3.1 Histoire

Autrefois, isoler du matériel dans des lieux tenus secrets comme par exemple Bletchley Park en Grande Bretagne, où se trouvait les « bombes » c.-à-d. les calculatrices de décodages, enfermer ce type de matériel dans des salles spécialisées où seules entraient quelques personnes possédant une clé et/ou une combinaison secrète, pouvait suffire pour éviter que des intrus, qu'ils soient inexpérimentés ou malintentionnés y accèdent. Par exemple *Enigma*, une machine construite par les allemands entre les deux guerres et utilisée par les troupes nazies pendant la seconde guerre mondiale pour crypter des informations militaires secrètes, était enfermée dans une valise spéciale fermée à clé dans un lieu d'accès restreint (bureau du chiffre, PC communication d'un sous-marin, salle radio d'un navire, etc.) ou transportée dans des conditions de sécurité particulières. Aujourd'hui aussi, des machines contenant des données sensibles d'une banque ou ayant de grandes capacités de calculs dans le domaine nucléaire sont installées dans des sites particuliers, éloignés des agglomérations, dans des enceintes bien gardées. Parfois l'emplacement même de ces sites est gardé secret.

Mais la protection physique a été très vite secondée par l'emploi de mots de passe. Cette combinaison de protection physique et de restriction logicielle est encore utilisée aujourd'hui. Ainsi, en vue de réaliser des économies d'échelle, on a pu permettre à un nombre plus grand d'utilisateurs d'une organisation

10. Voir par exemple le comportement des gardes dans *Coriolan*, acte V, scène II.

de travailler sur une même machine [11], où qu'elle soit, sans devoir distribuer des clés d'une salle spéciale. Ce type d'organisation pouvait apporter, outre une meilleure protection, une plus grande flexibilité. Plus tard, l'apparition des réseaux a permis de ne plus devoir se déplacer pour effectuer la moindre tâche informatique et d'améliorer encore le partage des ressources. De plus, en matière d'administration d'équipements mutualisés, pouvoir s'authentifier puis effectuer des tâches de gestion et de maintenance à distance est très utile surtout quand le dépistage, à temps, des anomalies de fonctionnement ou de problèmes de sécurité est impératif.

La multiplication des machines et des programmes, en particulier celle des programmes d'applications, a complexifié les usages : des utilisateurs de tous genres, informaticiens ou non, peuvent accéder à distance à plusieurs machines, voire à un réseau. Des équipements spécialisés dans le contrôle d'accès, comme les serveurs LDAP et les bases de données d'utilisateurs, font souvent partie intégrante des réseaux qu'ils soient locaux (entreprises, université,etc.) ou étendus comme les réseaux des grands opérateurs nationaux ou internationaux [12].

Pour des raisons économiques ou géographiques, des réseaux peuvent être juxtaposés. Leur exploitation nécessite souvent la présence de *passerelles*. On a un exemple d'une telle situation avec des réseaux nationaux. Parfois au contraire, des réseaux sont inclus dans d'autres, par exemple un réseau de laboratoire peut être un élément d'un réseau de campus, lui même élément de RENATER.

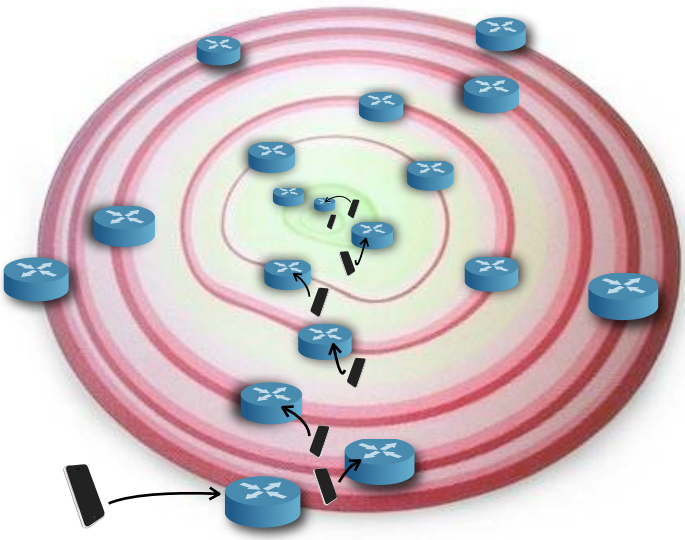

FIGURE 2.1 – Modèle en oignon

Dans ce paysage, deux type d'accès distants sont apparus : en *intranet* et en *extranet*. Cette distinction fait apparaître les prémices d'un modèle conceptuel

11. Des machines HPC comme SHARCNET ont des puissances de calculs qui autorisent leur utilisation par des centaines d'utilisateurs issus d'une dizaine d'universités.

12. Pour fixer les idées, le site OVH qui est l'un des plus importants hébergeurs européens relie à Internet sur son site plus de 30.000 serveurs.

extrêmement utilisé, celui d'une organisation en couches concentriques un peu comme les différentes couches d'un oignon (cf. FIGURE 2.1). Les couches les plus internes regroupent les équipements les plus fragiles et les machines les plus critiques qui appellent une protection plus forte que les éléments des couches extérieures. Pour réaliser le passage d'une couche à une autre en augmentant la protection sur la couche la plus interne, les ingénieurs utilisent des garde-barrières (cf. section 2.3.6).

### 2.3.2 Droits d'accès et domaine

**Périmètre de sécurité** Le paysage d'Internet n'a pas d'organisation claire, mais l'on peut dire sans trop se tromper qu'il est formé de réseaux, fonctionnant chacun comme un système autonome, et appartenant à une entreprise ou à un ensemble d'entreprises, privées ou publiques, ayant contracter des accords entre elles pour offrir un large choix de services aux utilisateurs, qu'ils soient individuels ou regroupés au sein d'entreprise. Ces services peuvent aller du simple raccordement physique entre les réseaux, au transport de données, à la mise à disposition de certaines machines [13], l'utilisation à distance de certains programmes, ou encore la consultation de certains fichiers (pages web, mail, etc). Le *business model* suivi est largement variable.

Une entreprise, et par conséquent son réseau, représente un *domaine*. Ce n'est qu'en disposant de l'accès aux machines de ce domaine, qu'un utilisateur pourra profiter des services offerts par ces différentes machines. On parle alors d'autorisation définie comme la « cession, à une autre entité, d'une permission officielle à faire ou à être quelque chose » [44].

**Droits d'accès** Une fois identifié comme utilisateur légitime d'un domaine particulier, un utilisateur est autorisé à faire certaines actions au moyen des machines de ce domaine. On parle de ses *droits d'accès*. La liste de ces actions peut être tenue à jour dans un fichier [14], dans une base de données ou au moyen de programmes particuliers [15].

Dans un tel fichier, chaque entrée contient des informations sur l'identité de l'utilisateur (nom, prénom, adresse mail, lieu de travail,etc.), sur les moyens possibles pour l'authentifier, et sur les services auxquels il peut prétendre (lecture et modification de fichiers, lecture seule des fichiers de configuration du système ou de fichiers vidéo, exécution de certains programmes, bande passante d'un débit précis, configuration d'adresse IP, accès à Internet, etc.). Pour toutes sortes de services, payants ou non, les droits d'accès sont fixés au moment où l'utilisateur s'inscrit.

### 2.3.3 Identification contre Authentification

Selon le RGS [45] « l'authentification a pour but de vérifier l'identité dont se réclame une personne ou une machine (ci-dessus désignée par entité). Généralement, l'authentification est précédée d'une identification, qui permet à

13. Par exemple serveurs hébergés.
14. Par exemple, pour les domaines réduits à une seule machine, le fichier `/etc/passwd`.
15. Par exemple, le service *Yellow Pages* ou YP, plus tard appelé *Network Information Service* ou NIS, développé par Sun. Par exemple encore, au moyen d'un LDAP (*Lightweight Directory Access Protocol*).

cette entité de se faire reconnaître du système au moyen d'un élément dont on l'a doté. En d'autres termes, s'identifier consiste à communiquer une identité préalablement enregistrée, s'authentifier consiste à apporter la preuve de cette identité ».

L'authentification est donc fondée sur le contrôle d'un élément que l'impétrant est le seul à posséder. Cela peut être une information [16], un savoir faire [17], un objet qu'il possède [18] ou une caractéristique physique. On appelle de tels éléments *éléments d'identification* ou *éléments d'accréditation* (en anglais *credentials*). Elle peut en effet se baser sur des *méthodes* différentes : par identifiant (en anglais *login*) et mot de passe, au besoin à utilisation unique [19], en utilisant la biométrie ou en utilisant une clé cryptographique [20]. Ces méthodes sont fort anciennes. Elles ont d'abord été utilisées pour l'accès local à une machine puis ont évolué pour faire face aux problèmes particuliers apparus avec la demande d'accès distant.

### 2.3.4 Authentification par mot de passe statique

C'est la forme la plus simple, la plus ancienne et la plus courante des méthodes d'authentification [21]. Sa mise en œuvre a évolué avec les découvertes successives de failles de sécurité. Sur les premiers systèmes Unix, un programme de connexion demandait à l'utilisateur de donner son identifiant puis son mot de passe qu'il comparait à des valeurs stockées à l'avance [22].

Il est à cet égard intéressant de voir un problème simple de sécurité et les méthodes mises en œuvre pour le résoudre. La tendance à l'économie de l'être humain ainsi que son manque de mémoire amenaient les utilisateurs à choisir des mots de passe *faibles*, trop courts ou en rapport avec leur nom, prénom, ceux de leurs proches ou à des dates aisées à connaître. Il devenait alors simple de se connecter par essais successifs pour peu que l'on pût en tenter suffisamment. Une réponse technique a alors consisté à exiger des mots de passe plus longs, combinant lettres et chiffres, en majuscules et en minuscules augmentant ainsi le nombre de mots de passe possibles. Parallèlement, on a limité le nombre d'essais autorisés. Mais à ces éléments techniques on a souvent ajouté des éléments psychologique par exemple en expliquant aux utilisateurs qu'ils pouvaient construire des mots de passe difficiles à deviner en utilisant par exemple les premières lettres des mots d'une phrase simple à mémoriser.

Mais cela n'a pas empêché des intrus d'établir des listes de mots de passe préalablement hachés et de parcourir le fichier de mots de passe du système pour trouver des correspondances. Elles sont d'autant plus importantes que les mots de passe sont faciles à deviner comme expliqué précédemment. Pour remédier à ce problème, Morris et Thompson (1979) ont décrit une technique où le système

16. D'Artagnan obtient l'aide de relais anglais au moyen du mot « forward ».
17. Dans l'Odyssée, Ulysse, méconnaissable, s'authentifie en bandant son arc resté à Ithaque et en l'utilisant, ce qu'aucun des prétendants ne sait faire.
18. D'Artagnan se fait reconnaître par Buckingham en lui présentant une bague.
19. Il s'agit d'un mot de passe qui change à chaque nouvelle authentification – en anglais *one time password* ou OTP.
20. César utilisait déjà la cryptographie il y a plus de 2000 ans et l'on peut raisonnablement penser qu'il n'était pas lui même très novateur.
21. Le mot de passe était déjà utilisé par les troupes de pharaon pour entrer dans les camps.
22. Très vite, on a découvert qu'il fallait *hacher* les mots de passe avant de les stocker et n'a guère fallu de temps pour se rendre compte que même cette précaution était inopérante et qu'il fallait trouver une solution de sécurité plus efficaces.

choisi un nombre aléatoire à chaque nouvelle identification, le concatène au mot de passe rentré par l'utilisateur et applique la fonction de hachage sur le résultat.

Ce procédé, utilisé pour une authentification locale, a été ensuite repris dans la conception des protocoles d'authentification à distance.

### 2.3.5 Authentification par mot de passe dynamique (OTP)

Toujours afin de résoudre les problèmes de sécurité de l'authentification présentés ci-dessus, certains systèmes demandent aux utilisateurs de changer fréquemment de mot de passe. Dans le cas extrême, un mot de passe différent peut être demandé à l'utilisateur à chaque nouvelle authentification. Il est utilisé une seule fois et immédiatement invalidé. Une liste de ces mots de passe est remise à l'utilisateur lors d'une entrevu face à face au moment de l'ouverture de ses droits. Cependant, l'existence d'une telle liste qui peut être égarée, perdue ou volée est en elle même une faille de sécurité.

Mais la technique du mot de passe à utilisation unique (en anglais *One Time Password* ou OTP) – aussi mot de passe dynamique – peut être mise en œuvre en remettant à l'utilisateur lors d'une entrevue face à face consécutive à l'ouverture de ses droits un objet, appelé clé OTP. Cet objet, qui n'est pas reproductible, affiche pendant une durée assez courte un code que l'utilisateur doit employer pendant cette durée comme élément de son authentification. Ainsi pendant chaque tranche de temps l'utilisateur sait quel est l'élément d'authentification en cours.

De cette façon, le système authentifiant n'a pas besoin d'enregistrer la liste de mots de passe de chaque utilisateur. De plus un adversaire ne peut pas deviner le prochain mot de passe à utiliser. OTP a été adoptée par de grandes entreprises (TF1) ou des administration (Min. Ed. Nat.) permettant ainsi d'accéder à distance à des programmes à travers les réseaux de leurs organisations.

### 2.3.6 Garde-barrières

Un garde-barrière est en matière de sécurité des réseaux l'équivalent *virtuel* de ce qu'est en matière de sécurité des bâtiments un mur ou une porte coupe-feu. Son rôle est d'empêcher ou de retarder la propagation d'un incendie d'une partie d'un bâtiment à une autre. C'est de cette analogie qu'il tire sa dénomination la plus répandue de pare-feu (en anglais *firewall*). Dans le domaine des réseaux, un pare-feu est placé, à l'initiative du responsable de sécurité d'une zone, entre deux zones différentes : l'une qu'il protège et qui représente par exemple un réseau d'une entreprise ou d'une institution ayant un domaine administratif bien délimité, nous l'appelons réseau intérieur ; et l'autre n'appartenant pas à ce domaine, donc hors du contrôle et source de dangers et qui est le plus souvent le réseau Internet, nous l'appelons réseau extérieur.

Pour remplir son rôle de protection, le pare-feu inspecte le trafic entrant dans la zone qu'il protège ainsi que celui en sortant. Il laisse passer certains paquets et en bloque d'autres. Pour cela, il applique un corpus de règles décidées à l'avance par les administrateurs du réseau intérieur et configurées dans une liste appelée souvent, par abus de langage, *Politique de filtrage*. Certains pare-feux sont aussi capables d'observer le trafic et de construire ensuite un ensemble de règles à appliquer, toujours modifiables ou adaptables ultérieurement par les responsables de la sécurité.

Un pare-feu peut se présenter simplement sous forme d'un programme installé sur une machine personnelle et ne s'occupant que du trafic la traversant [23]. La zone à protéger se réduit alors à cette seule machine. Le plus souvent on trouve au contraire les logiciels pare-feux installés sur des équipements dédiées au réseau, en général des routeurs, et protégeant ainsi de larges pans d'un réseau intérieur. Par métonymie, on appelle ces équipements pare-feux. D'ailleurs, ils ne contiennent pas en général seulement des programmes de pare-feux. Il peut s'y trouver également un programme assurant la translation d'adresses du réseau intérieur, si celles-ci sont privées, en adresses publiques (en anglais *Network Address Translation* ou NAT), un programme de routage, un programme de mise en place de réseaux privés virtuels (en anglais *Virtual Private LAN* ou VPN) si le réseau intérieur doit être déployé sur plusieurs sites de l'entreprise ou de l'institution qui sont éloignés les uns des autres, etc.

Les règles de filtrage sont un élément essentiel de la politique de sécurité d'un réseau. Elles permettent de limiter l'accès de l'intérieur vers l'extérieur et *vice versa*. Dans certains cas elles peuvent même permettre de limiter l'accès à une seule partie du réseau intérieur. On peut utiliser plusieurs critères de filtrage selon :

- les adresses IP source et/ou destination,
- l'interface du pare-feu où le flux est reçu (intérieure ou extérieure),
- le type de services c.-à-d. protocoles de la couche application par examen des numéros de ports des messages TCP ou UDP,
- les types de messages ICMP,
- l'utilisateur qui est à l'origine du flux, etc.

Les garde-barrières les plus simples appliquent un modèle markovien du trafic. Ils considèrent chaque message comme indépendant du reste du flux. On parle de pare-feux sans états (en anglais *stateless firewall*). Chaque règle prend donc la forme d'une suite de conditions basées sur un ou plusieurs des critères ci-dessus suivi de l'action qui devra être appliquée aux paquets vérifiant toutes ces conditions. Typiquement les actions sont : *Aucune*, qui sert simplement à noter le passage du paquet ; *Passer* [24], qui laisse passer le paquet sans autres traitement ; *Bloquer*, qui interdit le passage du paquet ; *Réinitialiser*, qui non seulement interdit le passage du paquet mais envoie de plus un message TCP/IP à sa source. Pour chaque paquet qui tente de traverser le pare-feu, la liste des règles est examinée et l'examen se termine à la rencontre de la première règle applicable. En fin de liste (c.-à-d. lorsqu'aucune règle applicable n'a été rencontrée) on applique implicitement l'une des deux actions *Passer*, c.-à-d. que tout ce qui n'est pas interdit est permis, ou *Bloquer*, c.-à-d. tout ce qui n'est pas permis est interdit. Naturellement le comportement de chaque type de pare-feux peut s'écarter plus ou moins du comportement ainsi décrit et il importe de ce reporter aux manuels pour les détails.

Pour donner de la fluidité au trafic et au même temps analyser plus profondément sa constitution, certains garde-barrières utilisent un modèle moins

23. C'est le cas des pare-feux intégrés dans certaines distributions de systèmes d'exploitation ou de logiciels spécialisés tels que ZoneAlarme.

24. En ce qui concerne l'action *Passer*, un certains nombres d'options permettent une collecte plus ou moins complète des éléments de ce paquet. Les données ainsi recueillies pourront faire ultérieurement l'objet de traitement, par exemple une étude de volumétrie, par exemple encore l'analyse de certaines attaques en vue d'en éviter la reproduction de type déni de service.

simpliste du trafic. On les appellent pare-feux à états (en anglais *stateful firewall*) en prenant en compte la logique de succession des différents messages d'un protocole donné, ils organisent un *droit de suite* laissant par exemple passer les réponses à un message dont ils ont choisi de laisser passer la demande. Cette examen peut être fait par exemple pour des protocoles de signalisation IP (ICMP) ou des protocoles de transport (TCP ou UDP).

Les pare-feux peuvent contenir une liste de contrôle d'accès (en anglais *Acces Cntrol List* ou ACL) où sont précisés les droits d'accès des clients à certains fichiers. D'autres ACLs décrivent aussi les ports et les adresses des réseaux ou des machines autorisés et interdits par le filtrage.

## 2.4 Outils cryptographiques et services de sécurité

Quand on parle de la sécurité de l'information, on pense habituellement à la cryptographie. Ce terme vient du grec $\kappa\rho\upsilon\pi\tau o\varsigma$ qui signifie caché et du verbe $\gamma\rho\alpha\phi\omega$ écrire. La cryptographie est en fait l'art des écritures secrètes. Généralement, on s'en sert pour garder secrète une information, pour qu'elle ne soit pas modifiée, ou pour authentifier une entité en vérifiant qu'elle connaît une information secrète. Le but de cette section est de donner un aperçu des méthodes cryptographiques les plus connues en vue de comprendre le contexte dans lequel il convient de les utiliser. Les principes suivis pour concevoir ces méthodes est hors du périmètre de cette thèse.

Du chiffrement par substitution qui consiste à remplacer dans un message certaines lettres par d'autres – toujours les mêmes [25], dont l'usage remonte au moins à Jules César, au chiffrement par transposition où l'ordre des lettres est transposé, à la méthode du masque jetable (*one-time pad*), et, enfin, aux méthodes d'exponentiation dans un groupe cyclique, l'histoire de la cryptographie est longue [26]. Selon Andrew Tanenbaum « Historiquement, quatre groupes ont utilisé l'art de la cryptographie et contribué à son développement : les militaires, les corps diplomatiques, ceux qui tiennent un journal intime et les amoureux »[46].

L'annexe A donne quelques éléments sur la façon dont un utilisateur peut choisir $K$ et construire une fonction $f_K : \mathcal{M} \longrightarrow \mathcal{C}$ définie sur un ensemble fini, mais très grand de messages et à valeur dans un ensemble de même cardinal de codons, puis déterminer $K'$ et $f_{K'}$ de telle sorte que la connaissance de $K$ ne donne aucune information sur la valeur de $K'$ et que $f_{K'}$ soit la réciproque de $f_K$. Le couple $(f_K,\ f_{K'})$ permet de construire une cryptographie asymétrique et $K$ et $K'$ s'appellent clés. Elle donne aussi des éléments sur la façon dont on peut construire une fonction $h : \mathcal{M} \longrightarrow c$ de l'ensemble des messages vers un ensemble beaucoup plus petit mais encore très grand de condensats de telle façon que si deux messages diffèrent peu [27] leur condensats diffèrent beaucoup,

25. Ce type de chiffrement, bien que peu solide, était encore utilisé pendant la guerre de sécession. On le réalisait alors au moyen d'un objet portant en regard deux alphabets et que l'on appelait *Cypher Disc*.

26. Sur l'histoire de la cryptographie depuis l'Égypte antique jusqu'aux années 1960, le livre *The Codebreakers - The Story of Secret Writing* de David Kahn est une référence instructive et d'abord aisée.

27. D'une lettre, d'un chiffre, par une permutation de lettres, etc.

et ainsi construite qu'étant donné un message $M_1$ il soit extrêmement difficile [28] de trouver $M_2$ tel que $h(M_1) = h(M_2)$. On dit qu'il est difficile de trouver des *collisions*.

Il existe essentiellement deux procédés cryptographiques. Le premier est le *chiffrement* (on dit aussi parfois avec une certaine imprécision *codage* ou *cryptage*) que l'on traduit en anglais par *ciphering*. Il s'agit d'effectuer une transformation sur un message (donnée sous forme de texte, image, vidéo, etc., programme de différentes sortes) dit *texte en clair* (aussi texte source), à l'aide d'une *clé de chiffrement* pour obtenir un message apparemment illisible dit *texte chiffré* de façon telle que le retour au message initial soit facile pour ceux qui possèdent une *clé de déchiffrement* et très difficile pour ceux qui n'en possèdent point. On dit dans ce cas que le chiffrement est sécurisé. Remarquons qu'il s'agit d'une opération bijective.

Au contraire, le *hachage* (en anglais *hashing*) consiste à produire à partir d'un texte source un résumé ou empreinte du texte, on dit aussi *condensat*, en général beaucoup plus court, mais conçu de façon telle qu'il soit extrêmement difficile de produire un autre texte ayant la même empreinte. Ainsi le hachage n'est pas une opération bijective. On qualifie ainsi les algorithmes de chiffrement par *réversibles* et ceux de hachage par *irréversibles*. Pour exprimer que la difficulté des tâches ci-dessus est suffisante, on dit que l'on utilise des algorithmes de qualité cryptographique, ou plus brièvement que ce sont des algorithmes cryptographiques. Le but de cette section n'est ni d'étudier leurs règles de conception, ni d'examiner l'ensemble des attaques à prendre en compte lors de cette conception. Il est plutôt de considérer leurs fonctions et leur performances afin de pouvoir choisir plus tard celui ou ceux qui conviendront à notre problème.

Nous verrons que selon le type de clés, le type d'algorithme ou d'association d'algorithmes, on réussit à cacher l'information dans un but déterminé ou contre un danger précis. Dans chaque cas, on dit que la cryptographie offre un service. Ces services constituent en majorité des réponses aux problèmes de sécurité énoncés plus haut : l'authentification en vérifiant que l'entité inconnue connaît une information secrète, la confidentialité en maintenant secrète une information, l'intégrité en empêchant la modification d'une information et la non-répudiation ou l'irrévocabilité en empêchant qu'une entité puisse nier avoir émis une information donnée.

### 2.4.1 Algorithmes de chiffrement

Le chiffrement cache (en anglais *encrypt*) un texte en clair (en anglais *plaintext*) noté traditionnellement $P$ en le transformant en texte chiffré (en anglais *ciphertext*), noté $C$, c.-à-d. écrit en chiffres par l'application d'un algorithme de chiffrement (en anglais *encryption algorithm*), noté $E$. Le chiffrement est une opération bijective et son inverse est le déchiffrement (en anglais *deciphering*), noté $D$. Ainsi, on écrit $C = E(P)$ et $P = D(C)$. Il se distingue de cette sorte de *codage* où chaque mot du texte en clair est remplacé par un symbole. Ce dernier procédé n'est plus utilisé pour cacher le sens d'un message, surtout depuis l'invention d'ordinateurs dont les capacités de transformation surpassent les capacités humaines autrefois indispensables.

28. Par exemple cela peut prendre plusieurs dizaines de siècles avec des moyens de calcul puissants.

En vue de garantir la dissimulation des textes chiffrés, on gardait autrefois les algorithmes secrets. Cependant une lente mais irrésistible évolution a amené à séparer le procédé en un algorithme et une clé, l'algorithme étant d'une qualité telle que seule la détention de la clé permet le chiffrement (ou le déchiffrement) du message. En effet, les efforts nécessaires pour modifier les algorithmes ou pour en inventer d'autres, lorsque ceux-ci sont *cryptanalysés* (on dit aussi *cassés*), étaient importants. De plus, l'idée que l'adversaire ne connaît pas le fonctionnement de l'algorithme et qu'il lui est par conséquent difficile de le découvrir ne sert qu'à donner des illusions aux concepteurs. On s'est même rendu compte que la publication des algorithmes en attirant les efforts de la communauté scientifique est un facteur de sécurité. C'est le principe de *Kerckhoff* : tous les algorithmes doivent être publics, seules les clés sont secrètes.

D'autre part, s'il y a lieu de communiquer une information sensibles d'une entité à une autre, ce caractère public évite d'échanger préalablement des algorithmes secrets de tailles assez importantes puisqu'ils sont connus de toutes les parties. On évite ainsi la réelle difficulté d'un échange de données à la fois volumineuses et sensibles puisqu'il suffit grâce à l'usage des algorithmes publics à clés secrètes, d'échanger seulement les clés.

Nous n'exposerons pas la palettes d'attaques connues et répertoriées contre les algorithmes de chiffrement renvoyant pour cela à [47]. Nous retiendrons simplement que l'efficacité de ces attaques varie selon la longueur des clés, l'algorithme considéré voire même la façon dont cet algorithme est effectivement mis en œuvre. Dès lors, il devient possible de définir une solidité des processus de chiffrement, en mesurant par exemple le temps nécessaire à la découverte, en présence d'un texte chiffré, du texte en clair originel : plus un algorithme est solide, plus ce temps de décryptage [29] devient long. On peut même classer les procédés par solidité croissante.

Cette solidité est appréciée. Elle se paye souvent au prix de temps de calculs, de consommation d'espace mémoire ou d'un rallongement qui peut être important des messages à transmettre. Le choix de l'algorithme de chiffrement peut donc être orienté par les capacités non seulement des équipements qui chiffrent mais aussi des équipements qui déchiffrent. Ce point est important à retenir quant au choix ultérieur pour sécuriser les échanges dans les réseaux ad hoc.

Il existe deux grandes classes d'algorithmes de chiffrement : ceux à clé symétrique dit aussi *symétriques* et ceux à couple de clés dit *asymétriques*. Un algorithme de la première classe emploie une même clé pour les opérations de chiffrement et de déchiffrement, tandis qu'un algorithme appartenant à la deuxième classe se sert d'un couple de clés publique/privée, quand l'une chiffre l'autre déchiffre. On désigne aussi la clé symétrique et la clé privée par *secret* car elles ne doivent pas être divulguées contrairement à la clé publique.

**Algorithmes de chiffrement symétrique** La machine Enigma mettait déjà en son temps en œuvre un algorithme à clé symétrique : en vertu de la continuité des circuits électriques, si la touche A allumait la lampe W, la touche W allumait la lampe A.

29. C'est le contraire de *déchiffrement* qui consiste selon Larousse [41] à « rétablir dans sa forme primitive un texte chiffré en utilisant en sens inverse le procédé de transformation adopté par le chiffreur et connu du déchiffreur » et qui dans notre cas suppose que la clé de déchiffrement est connue.

Déjà considérée par Claude Shannon, utilisée au cours de la guerre froide pour le célèbre téléphone rouge et implantée dans les premières versions du système UNIX par la commande `crypt`, la méthode du ou exclusif (en anglais `XOR`) est l'une des plus ancienne. On sait que sous certaines conditions, elle est optimale. Elle impose la pré-connaissance par les parties désirant communiquer de très longues clés, ce qui la rend difficile à mettre en œuvre.

Pour cette raison, d'autres algorithmes ont été proposés. Par ordre chronologique, on peut citer DES (*Data Encryption Standard*) conçu par IBM et standardisé en 1977, Triple DES (noté aussi 3DES) publié également par IBM en 1999, Rijndael du standard AES (*Advanced Encryption Standard*) adopté en 2001 par le NIST (*National Institute of Standards and Technology*) qui sont les trois algorithmes symétriques les plus utilisés aujourd'hui.

On sait, depuis la découverte d'un certain nombre d'attaques que l'on peut mener pour en casser les clés, que DES est fragile. En matière de chiffrement, cette méthode qui est encore disponible sur de vielles machines peut encore éviter les plus grossières des indiscrétions. Triple DES, dérivé du DES, exécute celui-ci trois fois successives en utilisant plusieurs clés différentes. Enfin AES, qui a été conçu pour parer les attaques trouvées contre DES et 3DES, assure une meilleure sécurité, en particulier en permettant l'utilisation de clés plus longues.

**Algorithmes de chiffrement asymétrique** Parmi les algorithmes asymétriques, RSA, du nom de ses créateurs *Rivest*, *Shamir* et *Adleman*, qui coexiste avec d'autres algorithmes, est de loin, le plus répandu. Conçu autour d'un problème de théorie des nombres clairement identifié [30] et jugé difficile par la communauté mathématique, il garantit pour cela une qualité et une sûreté qui le font considéré comme un archétype. Nous donnons quelques éléments mathématiques en annexe B.

Les algorithmes de ce type ont été essentiellement créés pour palier le problème de partage de clé posé par les algorithmes à clés symétriques. La section suivante met en lumière cette difficulté et les solutions apportées.

Les deux types d'algorithmes permettent d'assurer les services d'authentification et de confidentialité. Cela est expliqué plus loin dans ce chapitre. Si RSA est réputé pour sa solidité, il demande au chiffrement comme au déchiffrement un volume de calculs non négligeable, tandis que les algorithmes symétriques, qui sont connus pour être moins solides, sont d'exécution plus rapide. Nous ne détaillerons pas les diverses méthodes et critères utilisés par les cryptanalystes pour démontrer qu'un algorithme est solide. Mais nous mettrons l'accent sur l'importance de la longueur de la clé pour établir cette solidité. En effet, des chiffrements faibles avec un algorithmes donné peuvent le devenir moins en augmentant la longueur de la clé, puisqu'une fois la force de l'algorithme établie, la découverte de la clé, seule, représente un danger.

**Choix de la longueur des clés de chiffrement** Les algorithmes de chiffrements gagnent en solidité par l'utilisation de longues clés c.-à-d. des clés ayant un nombre de bits assez élevés. Cette longueur est bien entendu relative au type de l'algorithme considéré. Alors qu'une longueur n'excédant pas 300 bits

30. Il s'agit de factoriser un grand nombre entier dont on sait qu'il est lui même le produit de deux grands nombres entiers premiers.

pourrait être suffisante avec un chiffrement symétrique, une longueur du module RSA d'au moins 2048 bits, soit 616 chiffres en numération décimal, est nécessaire dans le cas d'un chiffrement asymétrique. Ceci est dû à la présence de la clé publique, connue au moins de l'adversaire, reliée à la clé privée par un procédé mathématique, lui aussi connu puisque l'algorithme est public et donc exploitable lors du décryptage [48].

L'importance de la longueur de la clé s'explique par le fait qu'en l'absence de tout autre mécanisme plus facile pour décrypter, l'attaquant peut se replier sur la solution de l'*attaque par force brute* qui consiste à essayer toutes les clés possibles jusqu'à trouver celle qui convient. Il en découle que pour une longueur de clé égale à $l$, le nombre maximal d'essais est égal à $2^l$. La charge de travail de l'attaquant augmente donc de façon exponentielle quand la longueur de la clé croît.

Inutile de rappeler par ailleurs que les temps de chiffrement et de déchiffrement sont aussi affectés par le rallongement de la longueur de la clé et non pas seulement les temps de décryptage, puisque, comme nous l'avons expliqué plus haut, l'on perd en temps de calculs quand on gagne en solidité. Il apparaît donc qu'un compromis est à établir lors du choix de la clé. Il doit tenir compte de la sensibilité de l'information à cacher et des risques encourus : un message d'ordre militaire est bien plus sensible qu'un message échangé entre journalistes [31]. Les performances des processeurs utilisés pour chiffrer ou, par l'attaquant, pour décrypter sont aussi à prendre en considération. Or celles-ci ne cessent de s'améliorer, conséquence de la capacité des processeurs à inclure de plus en plus de transistors, phénomène modélisé depuis longtemps par la loi empirique de Moore.

**Limites du chiffrement** Il y a des protections que le chiffrement ne peut pas garantir, par exemple qu'un texte ne soit pas modifié. En effet une personne malintentionnée est capable de changer le texte chiffré même si elle ne connaît pas sa version en clair. Cela peut être facilement détectable si le texte source est en caractères mais l'est moins s'il est en binaire. Pour faire en sorte qu'une machine puisse le détecter, on protège l'intégrité des informations grâce à des algorithmes cryptographiques de *hachage*.

### 2.4.2 Hachage cryptographique

Le hachage cryptographique prend un texte en clair de longueur indéterminée et le transforme en texte chiffré de longueur fixe, généralement beaucoup plus petit que le texte original, et appelé *hach* (en anglais *hash*) ou *condensat de message* (en anglais *message digest*). Contrairement aux fonctions de *checksum* qui servent à vérifier qu'une information reçue n'a pas subit de modifications suite à des intempéries de transmissions ou d'enregistrement, les fonctions de hachage, aussi algorithmes de hachage, servent à vérifier qu'une modification délibérée de l'information n'a pas eu lieu. De plus, elles ont été conçues pour qu'il soit quasiment impossible de retrouver le texte en clair à partir du hach, à l'inverse du checksum.

31. En 2000, une clé de 56 bits suffisait pour un usage à domicile, une clé de 128 bits suffisait pour un usage commercial, et celles de 256 bits suffisait pour un usage par l'état.

En plus de protéger l'intégrité des informations, le hachage est utilisé pour stocker les mots de passes de façon chiffrée afin qu'un intrus ne puisse pas les utiliser (cf. section plus haut). Un algorithme de chiffrement pourrait aussi faire l'affaire, mais l'exécution des algorithmes de hachage est connue pour être plus rapide que celles des algorithmes de chiffrement. On utilise aussi le hachage pour envoyer les mots de passes à travers le réseau.

Au même titre que les algorithmes de chiffrement, les algorithmes de hachage sont publics. Il est donc à noter qu'il ne faut pas envoyer un hach au même temps que son texte en clair, sinon il suffirait de changer chacun des deux pour qu'à la réception l'on ne puisse pas détecter la modification.

Enfin, certains algorithmes de hachage se servent de clés cryptographiques. C'est par exemple le cas de HMAC (à développer).

### 2.4.3 Services de sécurité

La cryptographie rend quatre services essentiels : l'authentification, la confidentialité, l'intégrité et la non répudiation. Elle permet aussi de se prémunir contre certaines attaques comme l'attaque par rejeu. Nous montrerons dans cette section comment les divers types d'algorithmes exposés précédemment réalisent ces fonctions.

#### 2.4.3.1 Authentification cryptographique

L'authentification (cf. section 2.3.3) cryptographique utilise des mécanismes de cryptographie et est basée sur la connaissance d'une clé cryptographique préalablement partagée entre l'entité qui authentifie ou *authentifiant* (en anglais *authenticator*) et l'entité qui s'authentifie ou *client* (en anglais *supplicant*). Au cours de l'authentification, la clé n'est pas envoyée telle quelle : un nombre aléatoire (en anglais *random*, noté $R$), appelé *défi* (en anglais *challenge*), est choisi par l'*authenticator* et chiffré par l'authentifié. Pouvoir chiffrer avec la clé secrète apporte ainsi la preuve que l'*authenticator* est bien l'entité qui s'était identifiée. On dit que ce protocole d'authentification est du type *question-réponse*.

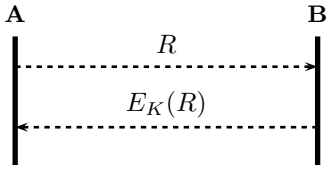


FIGURE 2.2 – Authentification cryptographique

#### 2.4.3.2 Confidentialité des données

La confidentialité est le fait de « garder une information secrète à tous sauf à ceux qui sont autorisés à la voir » [44]. Elle consiste à garantir que nulle autre entité que le destinataire d'un message ne puisse en prendre connaissance. Si un chiffrement symétrique est utilisé, il suffit d'employer la clé de chiffrement. Au contraire, si un chiffrement asymétrique est utilisé, il y a lieu de voir qu'il ne peut être effectué qu'avec une clé publique, car seule l'entité possédant la clé

privée correspondante peut déchiffrer. La confidentialité est ainsi préservée. Autrement, un chiffrement avec la clé privée impliquerait que toute entité pourrait le déchiffrer avec la clé publique.

#### 2.4.3.3 Intégrité des données

L'intégrité est le fait d'« assurer qu'une information n'a pas été altérée par un moyen non autorisé ou inconnu » [44]. La garantie d'intégrité consiste en l'impossibilité de modifier délibérément une information qui doit être protégée. Les algorithmes de chiffrement peuvent concourir à ce service, mais ne sont pas aussi efficace que les fonctions de hachage, un condensat de messages étant en général bien plus court qu'un message chiffré.

#### 2.4.3.4 Non répudiation

La non répudiation est le fait d'« empêcher le déni d'engagements ou d'actions antérieurs » [44].Il s'agit d'assurer qu'une entité qui a effectué une action ou donné un ordre ne puisse pas se dédire plus tard de l'avoir fait. La cryptographie asymétrique permet de rendre ce service contrairement à la cryptographie symétrique.

#### 2.4.3.5 Absence de rejeu

Certains attaquants cherchent à comprendre les actions que peuvent provoquer un texte chiffré une fois utilisé dans un système sans pour autant comprendre son contenu ou essayer de le déchiffrer. Il se contente d'observer ses effets et au mieux de deviner sa signification. Il suffit alors de le capturer et de s'en servir dans le même contexte pour obtenir à nouveau la réaction attendue du système. Cette approche par boîte noire s'appelle l'*attaque par rejeu*. On souhaite évidemment l'éviter.

La cryptographie permet indirectement de remédier à ce problème. A cet effet, une information sur l'heure de l'envoie, le délai de validité, ou, dans le cas d'échanges de plusieurs textes chiffrés successifs, le numéro de séquence associé au numéro de l'envoi, peut être ajoutée au texte en clair et chiffrée au même temps que lui. Ainsi le système qui déchiffre peut vérifier la cohérence de l'heure ou de la successions des messages et détecter toute tentative de rejeux.

Ces cinq protections assurées par la cryptographie sont celles de base dont on tient habituellement compte dans toute conception de système. Bien sûr, d'autres protections peuvent être envisagées mais en tout cas elles ne doivent être prises en compte que si le danger est réel c.-à-d. que la probabilité que cela puisse arriver n'est pas négligeable.

## 2.5 Relation de confiance et gestion de clés

Que ce soit à travers les techniques de contrôle d'accès ou les techniques cryptographiques, nous nous intéressons dans cette thèse à la sécurité des échanges et des données dont ils impliquent le transfert entre les différentes parties. Celles-ci sont représentées ou bien par un réseau tout entier ou bien par des entités

individuelles. Quand il s'agit du réseau tout entier, ce sont en réalité les équipements réseau qui prennent en charge les échanges entre le réseau et d'autres entités.

Afin de mettre en œuvre ces techniques, en particulier l'authentification avec mot de passe, le chiffrement et le hachage, il est important pour les différentes parties de se mettre *préalablement* d'accord sur les éléments secrets qui serviront à les identifier ou à cacher leurs messages. Ici, il s'agit des mots de passe, des clés de chiffrement et de déchiffrement et/ou des clés de hachage. Parfois, face à la multitude des algorithmes, leurs divers degrés de solidité et les multiples services qu'ils peuvent rendre, ces éléments sont négociés au même temps que les algorithmes qui vont s'en servir. Cette phase établie donc une *relation de confiance* entre les différentes parties concernées grâce à un *partage de secrets* .

Il va de soit que ce partage nécessite un échange sécurisé où un adversaire ne doit pas être en mesure d'écouter, de voir ou d'enregistrer le secret, ou les éléments d'identification de manière générale. Le moyen le plus élémentaire est celui par contact physique de main en main. Là le secret est chuchoté, enfermé dans une enveloppe, tapé à la dérobée, etc. Le contact physique peut aussi se faire grâce aux ondes infrarouges entre deux entités séparées de quelques mètres de distance et en ligne de vue directe. Enfin à travers la fibre optique, il est possible d'échanger un secret à une distance de quelques kilomètres au moyen de la mécanique quantique. C'est un moyen prometteur mais pas encore tout à fait mûr.

A défaut de contact physique, on peut avoir recours à un contact *virtuel* par la mise en place d'un *canal sécurisé*. Cette idée est souvent basée sur la présence d'un *tiers de confiance* (en anglais *trust third party* ou *TTP*) en qui chacune des parties, client et authentifiant, ont confiance et, plus globalement, sur une *hiérarchisation des relations de confiance*, vu aussi comme une *chaîne de confiance*.

### 2.5.1 Hiérarchisation de la confiance

Dans la ligne de tradition de la cryptographie, supposons que deux entités *Alice* (A) et *Bob* (B) veuillent partager un secret sans qu'*Ève* ne puisse les espionner. Si A et B font déjà confiance à une entité tierce, que l'on désigne par *Big Brother* (BB), c.-à-d. que chacun a déjà partagé une clé avec BB, ils peuvent établir un canal sécurisé. Pour cela, plusieurs méthodes ont été mise en œuvre, par exemple celle utilisée par Kerberos. Dans ce système, BB est le serveur Kerberos, il choisi une clé qu'il chiffre avec sa clé partagée avec A et l'envoie à celle-ci et fait de même avec la clé partagée avec B. Dorénavant A et B partage cette clé et peuvent communiquer de façon chiffrée. Remarquons au passage que BB connaît aussi cette clé, et à moins qu'il ne soit une autorité supérieure, comme un organisme étatique, une banque, etc. il est difficile de lui faire confiance.

Les algorithmes cryptographiques à clés asymétriques ont été inventés principalement pour résoudre ce problème. Cela est d'autant plus problématique que le nombre d'entités qui doivent connaître le même secret est grand : le risque de divulguer ce secret croit quand le nombre d'entités augmente. Deux clés associées algébriquement sont utilisées dans le chiffrement asymétrique : une clé publique et une clé privée. Seule l'entité qui a choisi la valeur de la clé privée la connaît et la détient, alors que tout autre entité ne peut connaître que la clé

publique associé.

Supposons que A comme B possèdent chacun ce type de couple de clé, et BB aussi. Il suffit que BB certifie que A et B sont bien les détenteurs de leurs clés respectives pour que A puisse faire confiance à la clé publique de B et *vice versa*. Dorénavant, chaque fois que A veut cacher un message destiné à B, elle le chiffre au moyen de la clé publique de celui-ci et aussi, bien entendu, d'un algorithme asymétrique.

Certifier consiste à signer la clé publique d'une entité et éventuellement d'autres informations liées à son identité, par exemple son nom, son adresse mail, le lieu où elle travaille, etc. avec la clé privée de BB. La signature obtenue est mise avec la clé publique et le reste des informations dans un même fichier que l'on appelle *certificat*.

Une fois cela est fait, cette opération peut être effectuée à plusieurs niveaux c.-à-d. que A peut elle même certifier des clés publiques d'autres entités qui elles à leur tour peuvent certifier d'autres clés publiques. Ainsi une hiérarchie de certification est établie. Généralement, l'entité qui est au plus haut est une autorité réputée et celles qui sont plus bas le sont moins.

Nous constatons que cette méthode permet de partager la clé publique avec autant d'entités que l'on veut et garder malgré ça la clé privée secrète contrairement à la méthode utilisant des algorithmes symétrique ou des mot de passes où la clé secrète ou le mot de passe doivent être partagés avec autant d'entités qu'il est nécessaire. Mais la cryptographie asymétrique n'a pas que des bons côtés car bien qu'il est permis de chiffrer avec la clé publique, cette opération est coûteuse en temps est ne doit pas être employer pour chiffrer une grande quantité de données.

L'idée lumineuse qui a été trouvée consiste à chiffrer non plus ces données mais une clé symétrique choisie conjointement par A et B. Celle-ci peut ensuite servir à négocier d'autres clés symétriques à d'autres fins utiles. Le protocole IKE (*Internet Key Exchange*) permet de mettre en place ce type de négociation de clé symétrique en se basant sur l'existence préalable de relation de confiance entre A et B via BB, certificateur de leur clés publiques. IKE fait appel à des protocoles de négociation de clés comme *Diffie-Hellman*.

D'autres modèles de confiance utilisant les certificats existent, par exemple PGP (*pretty good privacy*) . Celui-ci utilise la notion de chaîne de certificats.

### 2.5.2 Renforcement de la vigilance en environnement distribué

Les modèles décrits ci-dessus sont certes convenables quand il s'agit de considérer une entité individuelle seule maîtresse de toutes sortes de chiffrement en émanant, par exemple un client d'une banque donne seul l'ordre de vente ou d'achat d'un produit en y envoyant un message chiffré. Mais que se passerait-il si ce genre de décision devait être pris par une société ? Serait-il judicieux de confier cette responsabilité à une seule personne ? L'affaire Kerviel nous a montré le contraire. Pour permettre à plusieurs entités de prendre chacune une part de responsabilité dans une prise de décision importante, et pour empêcher que tout autre entités ne puisse violer leurs identités, on peut utiliser la *cryptographie à seuil*.

Dans le système de cryptographie à seuil, la clé de chiffrement est partagée en plusieurs parts, chacune d'elles est donnée à un participants. Il suffit qu'il

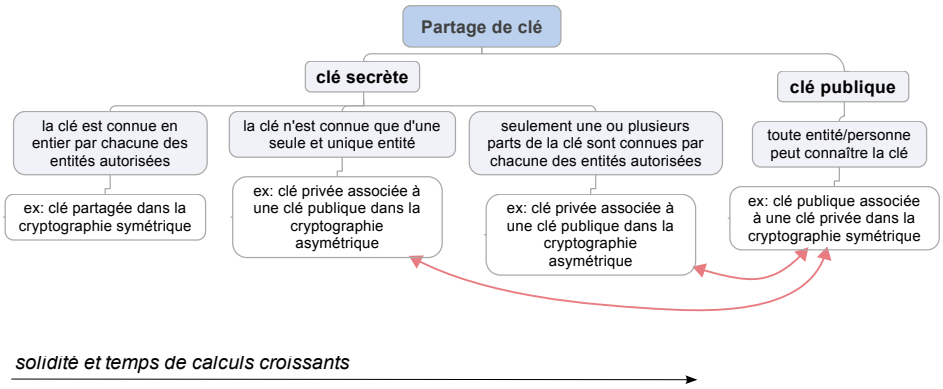


FIGURE 2.3 – Procédés de partage de clés cryptographiques

y est un nombre suffisant d'entre eux pour pouvoir retrouver la clé : c'est le *seuil cryptographique*. Cette techniques apporte une meilleure sécurité du fait de l'impossibilité de chiffrer sans l'apport d'un nombre seuil de parts de clé. Nous détaillerons aussi bien son fonctionnement que ces applications à la section 4.4.2 du chapitre 4 de cette thèse.

### 2.5.3 Partage de clés

Tout au long de cette section, nous avons montré que les techniques de partage sont nombreuses et indispensables à la mise en place de la sécurité. En plus des clés cryptographiques, nous avons vu que les mots de passe, lorsqu'ils sont employés, sont partagés entre au moins deux entités, par exemple un client et un serveur. Il ne nous semble pas utile de nous attarder sur la manière de les partager. Au fond ce partage est similaire à celui des clés de chiffrement symétrique. Ainsi, nous nous concentrerons sur le partage des clés cryptographiques en général et à chaque fois qu'il s'agira de partage de clé symétrique, on comprendra qu'il est aussi valable pour les mots de passes.

La figure 2.3 récapitule l'ensemble des techniques de partage de clés que nous avons exposées jusque là. Globalement, deux types de partage existent : le partage de clé secrète et le partage de clé publique. Ce dernier, comme son nom l'indique ne restreint pas l'usage de la clé au contraire du premier.

De plus, le premier type de partage dépend de la façon dont la clé est partagée. Elle peut être connue entièrement de chacune des entités concernées, deux ou plus, c'est le cas des clés symétriques. Elle peut n'être connue que d'une seule entité, celle qui l'a créée, c'est le cas des clés privées de la cryptographie asymétrique. Enfin elle peut être divisée en plusieurs morceaux de clé répartis entre plusieurs entités, celles qui y sont autorisées, chacune pouvant détenir un ou plusieurs morceaux au même temps. Ceci est le cas des clés utilisées dans la cryptographie à seuil. Shamir a été parmi les premiers à décrire un procédé se servant des polynômes pour calculer chaque morceau, appelé aussi part, de clé [49].

Sur la figure 2.3, ces méthodes de partage sont classées par ordre croissant de solidité. Meilleure est la solidité, plus complexe est le calcul cryptographique

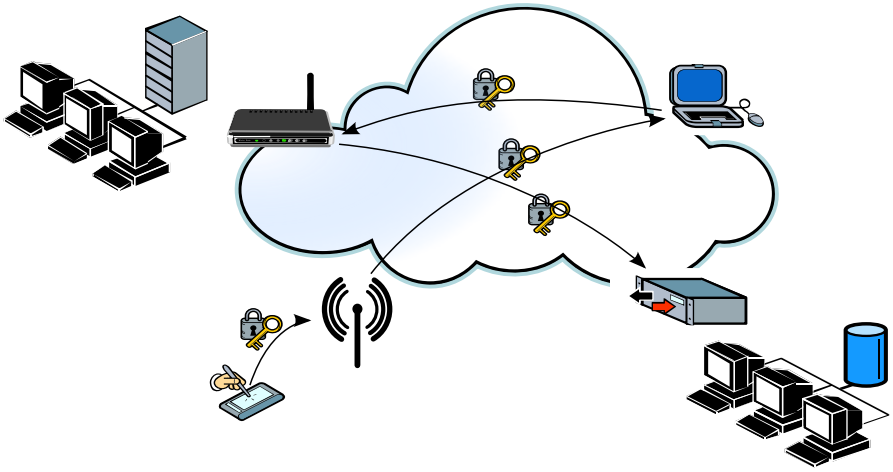

FIGURE 2.4 – Divers types de communications chiffrées (Les cadenas indiquent les communications chiffrées)

nécessaire, d'où l'importance de bien définir les besoins cryptographiques chaque fois qu'on y a recours.

## 2.6 Protocoles de sécurité

Une autre typologie repose sur la situation de l'initiateur de la communication. Il peut en effet s'agir d'un nœud intérieur ou extérieur au réseau (ou à la couche de réseau considérée). La sécurité d'un réseau comporte plusieurs aspects : la sécurité des communications point à point, un point étant une entité intérieure au réseau et l'autre étant une entité qui y est intérieure ou extérieure, et la sécurité des communications entre un point extérieur et le réseau (cf. FIGURE 2.4). Dans le premier cas, il suffit d'utiliser les techniques ci-dessus pour sécuriser les protocoles nécessaires à la communication points à point. Dans l'autre cas, les protocoles utilisés ne servent qu'à la mise en place de la sécurité des protocoles précédents, et ils impliquent en générale plusieurs parties.

Longtemps restreinte au premier type, la sécurité des échanges a évolué vers le second car face à la multitude d'applications, souvent fournies au sein du même réseau pour un utilisateur donné, on a pensé à créer un *service de sécurité* central pour toutes les applications. Ainsi, les bases de données, les équipements qui authentifient ou ceux qui négocient les clés cryptographiques sont mutualisés entre les diverses applications comme telnet, ftp, web, mail, etc. Celles-ci se contentent ensuite d'utiliser ces moyens mutualisés, en particulier pour chiffrer et hacher.

Un service de sécurité centralisé se présente sous forme de plusieurs machines dédiées au fonctionnement du réseaux et à sa sécurité. Des protocoles ont été conçus pour permettre à ces machines de communiquer et mettre ainsi en place la sécurité du réseau tout entier. Ces protocoles eux mêmes utilisent les techniques cryptographiques pour négocier les éléments des techniques cryptographiques utilisées par les applications. C'est en fait, une fois de plus un fonctionnement

hiérarchique, en d'autres termes en suite de protocoles. Par exemple IPsec (*IP security*) est une telle suite. Elle comporte les protocoles IKE, ESP, AH.

IPsec est souvent utilisé pour sécuriser les communications entre deux équipements du réseau. En effet il sert à mettre en place une association de sécurité servant plus tard à protéger toute sorte de communications y compris celles qui établissent d'autres associations de sécurité.

Pendant quelque temps, il y a eu une hésitation quant à la couche de la pile de protocoles de communication qui devait fournir les services de sécurité. Chiffrer les données au niveau deux c.-à-d. au niveau de la liaison de données aurait impliqué qu'à chaque fois qu'elles arrivaient à un routeur, celui-ci devait les déchiffrer pour pouvoir les router. Alors qu'au niveau de la couche application cela aurait amené à définir pour chaque application une protection différente. C'est ainsi que les experts de la sécurité ont proposé d'inclure les solutions de sécurité dans la troisième couche c.-à-d. la couche réseau en inventant la suite IPsec et dans la couche transport en inventant le protocole SSL (*Secure Socket Layer*) qui au terme d'une évolution devenu TLS (*Transport Layer Security*).

Ces deux protections ont de nos jours des applications dans le domaines des VPN (*Virtual Private Network*) et dans le domaine des infrastructures de contrôle d'accès et de comptabilité, désignées par l'acronyme AAA. Parfois les équipements qui mettent en place le contrôle d'accès et la comptabilité sont aussi capables de configurer des VPN.

Par ailleurs dans le cadre des réseaux d'accès sans-fil où généralement un ou plusieurs routeurs sans-fil mettent en place un réseau local, en vue de créer un environnement de transmission semblable à celui des réseaux filaires, des solutions de sécurité au niveau de la couche de liaison de donnée peuvent être implémenté. C'est le cas du protocole IEEE 802.1X qui repose sur le protocole définie par la norme IEEE 802.11 dans la couche deux et qui utilise la technologie WEP (aujourd'hui dépassée) pour protéger les communications dans la boucle locale.

## 2.7 Conclusion

Cette incursion dans le domaine de la sécurité des réseaux en a montré la complexité et l'étendu. L'absence de conception intégrée dès les fondations en est un élément déterminent. De plus, rien ne présageait du foisonnement et de la multiplicité des appareils informatiques que l'on connaît aujourd'hui, non plus de leur rapide évolution technique ni de leurs usages diversifiés. La sécurité des réseaux est un domaine très vaste qui concerne aussi bien le matériel que ses contenus comme les données, les programmes et les logiciels et ce qu'ils offrent comme les services.

Les techniques de sécurité autres que celles qui concernent le matériel reposent sur l'authentification des entités, utilisateurs et/ou équipements, le filtrage de trafic, le partage de relation de confiance et la cryptographie. Ainsi des protocoles d'authentification, de mise en place d'association de sécurité, de négociation de clés cryptographiques ont été créés. Ces solutions interviennent à divers niveaux de la pile de protocoles, c'est pourquoi ce domaine est qualifié de *transverse*.

Les dangers sont divers mais le fait de les répertorier et de les classer par ordre de gravité et de vraisemblance aide à la conception de solutions globales

adaptées en évitant des complications inutiles et en limitant les coûts au strict nécessaire.

Enfin, l'étude sommaire des réseaux à infrastructure qui vient d'être menée donne des pistes précieuses sur les méthodes et outils adaptés aux réseaux ad hoc que nous envisageons dans le prochain chapitre.

# Chapitre 3

# Taxonomie des services AAA dans les MANETs

## Sommaire


**3.1 Introduction . . . . . 48**
**3.2 Vue générale de AAA . . . . . 48**
3.2.1 Modèle d'architecture . . . . . 48
3.2.2 Protocoles . . . . . 50
3.2.3 Processus et mécanismes . . . . . 51
3.2.3.1 Inscription . . . . . 52
3.2.3.2 Initialisation . . . . . 52
3.2.3.3 Authentification . . . . . 52
3.2.3.4 Établissement de clés de session . . . . . 53
3.2.3.5 Autorisation . . . . . 54
3.2.3.6 Session autorisée . . . . . 54
3.2.3.7 Révocation / Isolation . . . . . 54
**3.3 Taxonomie . . . . . 54**
3.3.1 Modèle de dépendance . . . . . 55
3.3.1.1 MANET tributaire . . . . . 55
3.3.1.2 MANET autonome . . . . . 57
3.3.2 Modèle d'architecture . . . . . 57
3.3.2.1 Architecture centrale . . . . . 58
3.3.2.2 Architecture hiérarchique centrale . . . . . 58
3.3.2.3 Architecture hiérarchique distribuée . . . . 60
3.3.2.4 Architecture pair-à-pair . . . . . 60
3.3.3 Services commercialisables et comptabilité . . . . . 61
3.3.3.1 Types de services . . . . . 61
3.3.3.2 Méthodes de comptabilisation . . . . . 62
3.3.4 Modes de communication AAA . . . . . 64
3.3.4.1 Protocoles one-to-one . . . . . 64
3.3.4.2 Protocoles d'authentification one-to-many . 68
3.3.5 Synthèse . . . . . 68
**3.4 Considérations annexes . . . . . 70**

3.4.1 Bases de données . . . . . 70
3.4.2 Architecture matérielle et logicielle . . . . . 71
**3.5 Discussion . . . . . 71**
**3.6 Conclusion . . . . . 73**

## 3.1 Introduction

Par AAA (*Authentication, Authorization, Accounting*), on désigne un système de mécanismes, protocoles et architectures dont le terme, et donc le but est l'*accounting* c.-à-d. la tenue d'une comptabilité. Mais pour avoir une valeur, la comptabilité doit pouvoir s'appuyer sur des éléments factuels irrécusables. La piste d'audit doit permettre d'identifier qui a fait quoi. Il s'agit ensuite de garantir que les opérateurs du réseau puissent affirmer avec une certitude raisonnable que l'entité qui prétend faire des opérations et bien celle qu'elle affirme être. C'est le problème de l'*authentification*. Par ailleurs, cette identification doit être incontestable. C'est le problème de la *non répudiation*. Il s'agit ensuite de garantir qu'une entité effectuant des opérations *via* le réseau ait été explicitement autorisée à le faire. C'est le problème de l'*autorisation*

La présence d'enjeux financiers rend particulièrement sensibles les étapes du processus dont les défectuosités pourraient aller jusqu'à remettre en cause jusqu'à l'existence même du réseau. Il s'agit donc, en grande partie, de problèmes de sécurité.

## 3.2 Vue générale de AAA

Le terme AAA est apparue au milieu des années 1990 suite à l'effort de standardisation né de la coopération entre l'IEEE et l'IETF, à travers son groupe de travail AAA (*AAA Working Group*) [50], pour développer des applications AAA [51] [1].

### 3.2.1 Modèle d'architecture

La croissance du nombre d'abonnés aux services Internet a amené les FAIs [2] à déployer un nombre croissant de serveurs d'accès au réseau (en anglais *Network*

1. Notons aussi la participation de l'IRTF (*Internet Research Task Force*) à cet effort de standardisation à travers son groupe *AAA Architecture Research Group* [52] créé à la fin des années 1990 et dont le but était de définir une architecture et un modèle AAA interorganisationnels.

2. Un fournisseur d'accès à Internet ou FAI (en anglais *Internet Service Provider* ou ISP) est un organisme qui offrent un ensemble de services liés à l'utilisation d'Internet, alors qu'un opérateur de télécommunications est un organisme qui offre un service de communication à distance. Historiquement, les opérateurs de télécommunications fixes ont existé avant les FAIs. En installant des équipements spécifiques, appelés DSLAM, ces derniers ont pu exploiter les réseaux des opérateurs pour mettre en place des services Internet par ADSL. Souvent les opérateurs de télécommunications avaient eux-même installé leurs DSLAMs, comme par exemple l'opérateur historique France Télécom ou l'opérateur Bouygues. Pour cette raison, les mots FAI et opérateur sont souvent employés pour désigner un même organisme. Cependant, aujourd'hui, les techniques permettant d'offrir des services d'accès à Internet et/ou de télécommunications sont diverses et variées et des FAI peuvent exister sans avoir recours aux réseaux d'opérateurs. Parfois même, ils sont *virtuels* et ne possèdent, ni ne gèrent, un réseau pour offrir de tels services. C'est le cas notamment de DARTY.

*Access Server* ou NAS) [3]. Pour pallier un lourd travail d'administration de ces équipements, les FAIs ont mis en place des serveurs centraux, appelés *serveurs AAA*, contenant les outils, politiques et données nécessaires à l'authentification et l'autorisation des utilisateurs et à la comptabilisation des ressources consommées.

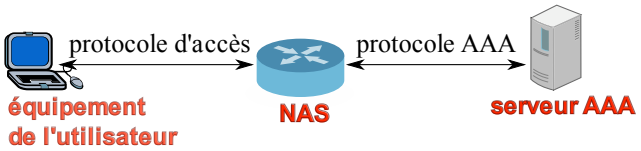


FIGURE 3.1 – Architecture AAA simple

Ainsi, lorsqu'un abonné accède au réseau de son opérateur, il est authentifié et autorisé par le serveur AAA *via* le NAS. Ce dernier collecte les informations sur les consommations de l'abonné et les envoie au serveur AAA. Les communications entre le NAS et le serveur AAA ont lieu grâce à un *protocole AAA*, celles entre l'équipement de l'utilisateur et le NAS grâce un protocole d'accès comme PPP (*Point-to-Point Protocol*) (cf. FIGURE 3.1).

L'hétérogénéité des équipements de réseau et de leurs fonctionnements, la variété des protocoles et des services proposés par les FAIs et la diversité des modèles commerciaux [4], en particulier par les offres de services en mobilité, sont parmi les facteurs qui ont contribué à la complexification des architectures AAA. Au delà de ces différences, un modèle abstrait permet de mieux comprendre leurs composants et les mécanismes qui les animent. Ce modèle couvre aussi bien les situations d'itinérance (en anglais *roaming*) [5], où un client mobile bénéficie de la couverture de réseau d'un opérateur différent de celui chez qui il s'est abonné, que les situations d'absence d'itinérance.

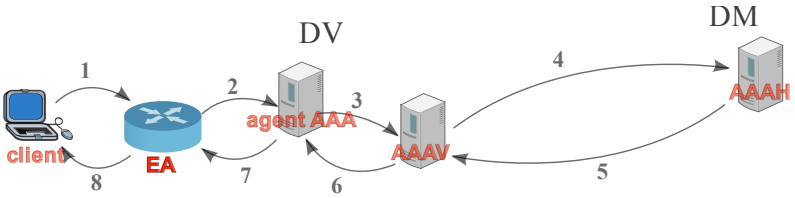


FIGURE 3.2 – Modèle des architectures AAA

La figure 3.2 montre le modèle suivi dans la conception des architectures

3. Utilisé par les FAI, le NAS est un équipement qui se trouve à l'interface entre le réseau téléphonique commuté et le réseau de données.Il transforme les communications téléphoniques en trafic de données IP.

4. En opposition au modèle scientifique, on entend par modèle commercial un *business model* c.-à-d. une schématisation du comportement type d'un ou plusieurs clients et d'un ou plusieurs fournisseurs lorsqu'un service est mis en jeu.

5. L'itinérance est une situation dans laquelle, par accord entre divers fournisseurs d'accès, un client donné peut, sous certaines conditions, bénéficier selon le lieu des services d'autres fournisseurs que celui qu'il a choisi.

AAA. L'opérateur où l'abonnement a été souscrit est appelé *opérateur mère de souscription* ou OMS. Son domaine de réseau est appelé *domaine mère* et noté DM (en anglais *home realm*). Le domaine de l'opérateur d'itinérance s'appelle *domaine visité* et est noté DV (en anglais *visited realm*). L'*équipement d'accès* au réseau, noté EA, se trouve à l'entrée du réseau d'accès entre l'équipement de l'utilisateur, ou *client*, et le DV. Remarquons qu'en l'absence d'itinérance, l'EA se trouverait à l'entrée du DM.

En fonction de la technologie d'accès, l'équipement d'accès peut être un NAS, une passerelle, un routeur, un point d'accès sans-fil, etc. Dans certains cas de protocole AAA (cf. section 4.2.2), il héberge un *client AAA* [6] capable de dialoguer avec le serveur AAA. Par exemple, suite à une demande d'accès émise par le *client*, l'EA communique avec un *serveur AAA* dans le DM, appelé serveur mère (en anglais *home AAA* ou AAAH). Cette communication peut traverser un ensemble d'agents et de serveurs AAA. En effet, des agents AAA [7] peuvent recueillir les demandes d'accès pour les envoyer ensuite au serveur AAA du DV, nommé AAAV (*visited AAA*). Ce dernier ne pouvant pas authentifier le client, inconnu car non abonné au DV, transmet, enfin, la demande au AAAH.

L'EA peut aussi héberger un *Enforcement Point* (EP). C'est une entité responsable de l'application de la politique de contrôle d'accès au réseau (cf. section 2.3 du chapitre 2). Lorsqu'un client fait une demande d'accès à une ressource ou à un service, ses droits sont examinés afin d'établir s'il peut y être autorisé. L'EP l'informe ensuite du résultat de cet examen.

Ce modèle d'architecture est souvent évoqué quand il s'agit de services d'accès à Internet fournis en mobilité [8]. Lorsque les MANET sont considérés comme une extension du réseau d'un FAI, les nœuds ad hoc peuvent utiliser les services AAA de ce FAI, donc une architecture AAA suivant le modèle décrit dans cette section, pour accéder au réseau. Cela sera décrit dans la section 3.3.1.

### 3.2.2 Protocoles

Un protocole d'accès [9] est utilisé entre le client et l'EA et un protocole AAA est utilisé dans les échanges entre l'EA et AAAH au sein de l'architecture AAA. L'EA joue, ainsi, un rôle d'intermédiaire entre le client et AAAH. Sur la figure 3.2, nous avons dessiné un seul échange, mais en réalité plusieurs échanges peuvent avoir lieu entre le client et AAAH à travers l'EA selon les spécifications des protocoles. Parmi les protocoles AAA connus, RADIUS [55] est encore utilisé par les opérateurs mais devenu de plus en plus inadéquat au fur et à mesure que leurs réseaux augmentaient de taille.

Son successeur, Diameter [56], répond mieux aux exigences de passage à l'échelle et s'adapte mieux au contexte de services en mobilité. RADIUS et

6. Dans le sens client/serveur AAA.

7. Il existe trois types d'agents AAA : agent de relais, agent de redirection et agent de translation. L'agent de relais s'occupe de router la demande d'accès. L'agent de redirection prévient le client AAA de la nécessité de diriger son trafic à une autre adresse si l'adresse du AAAV n'est plus valide. Enfin, l'agent de traduction traduit la demande dans un autre protocole si le serveur AAA n'utilise pas le même protocole que le client AAA.

8. Dans ce cadre, le protocole Mobile IP [53] [54] est utilisé à travers des technologies d'accès différentes telles que celles des réseaux cellulaires et des réseaux WLAN.

9. Un protocole d'accès peut par exemple être PPP (*Point to Point Protocol*) ou IEEE802.11, qui opèrent au niveau de la couche liaison de donnée ou PANA (*Protocol for carrying Authentication for Network Access* qui opère au dessus de la couche IP.) qui opère au niveau de la couche application. Tous les trois supportent des protocoles d'authentification.

Diameter utilisent, respectivement, un *client RADIUS* et un *client Diameter* au sein de l'EA. En revanche, Kerberos [57], un autre protocole AAA, peut nécessiter la présence d'un proxy Kerberos au niveau de l'EA. Kerberos est différent de RADIUS et de Diameter dans la mesure où il permet de construire une communication sécurisée avec une application par la mise en place d'une clé de session. Contrairement à Diameter qui a profité des retours d'expériences sur RADIUS, les aspects liés à la comptabilité et, en partie, à l'autorisation ne figuraient pas dans les premières conceptions de RADIUS et de Kerberos. Ils ont été ajouté progressivement [58].

Nous verrons dans ce chapitre qu'aussi bien RADIUS que Diameter et Kerberos ont été utilisés dans les MANETs pour mettre en place des services AAA. Leurs architectures ont été pour cela révisées et étendues. D'autres protocole AAA comme TACACS (*Terminal Access Controller Access-Control System*) [59] existent aussi mais, à notre connaissance, ils n'ont pas fait l'objet d'adaptation au contexte des réseaux MANETs.

### 3.2.3 Processus et mécanismes

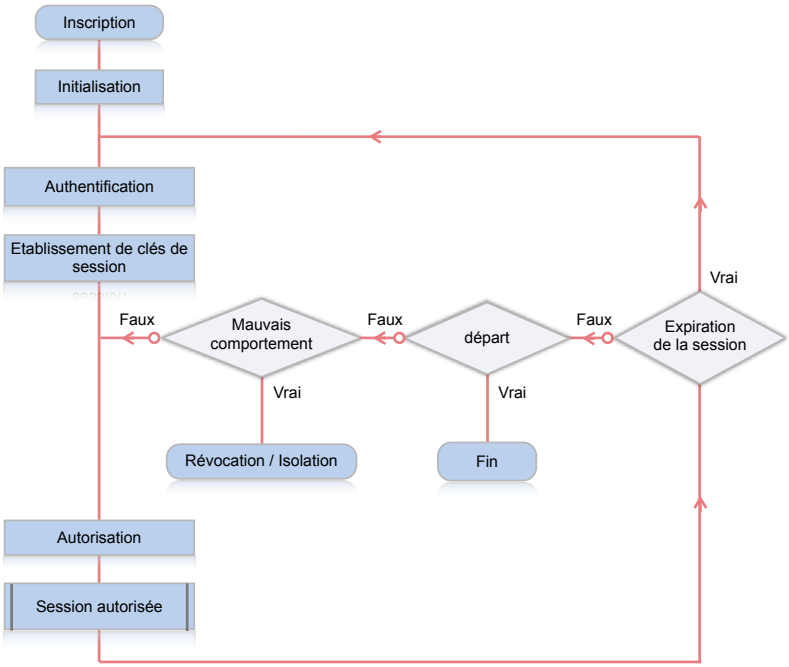


FIGURE 3.3 – Enchaînement des mécanismes AAA

Les processus AAA se déroulent selon les étapes illustrées par l'organigramme de la figure 3.3. C'est un organigramme simplifié et il ne saurait présenter chaque étape en détails.

#### 3.2.3.1 Inscription

L'*inscription* (en anglais *enrollment*) a lieu hors ligne à l'abonnement du client aux services d'un FAI. A cette étape, les différentes entités impliquées initialise une relation de confiance par la communication au client de ses éléments d'accréditation. A cette étape, des éléments supplémentaires peuvent être communiqués au client, par exemple l'adresse de l'EA, les adresses d'autres services de réseau comme ceux fournis par les serveurs DNS et DHCP, l'identité et les coordonnées du tiers de confiance [10], les algorithmes cryptographiques et les protocoles supportés par les différentes entités et dont l'utilisation sera faite pendant les étapes suivantes, etc. Certains opérateurs tels que Nerim [60] allouent en plus gratuitement une adresse IP définitive au client. Mais ce n'est en général pas le cas et l'adresse IP est plutôt allouée temporairement après authentification et autorisation du client. D'autre part, les droits d'accès sont aussi précisés et inscrits dans une base de données des clients ou une ACL (cf. section 2.3.2 du chapitre 1). A la fin de cette étape, on dit que le client est abonné aux services d'un opérateur.

#### 3.2.3.2 Initialisation

L'*initialisation* (en anglais *bootstrapping*) a lieu, en ligne, lorsqu'un client arrive à proximité d'un réseau dont il souhaite faire partie. Cela se traduire par l'exécution d'actions nécessaires à son fonctionnement dans le réseau. Parmi ces actions, on peut citer la découverte de nœuds voisins avec qui il sera amené à communiquer directement, la configuration d'une adresse IP selon la règle précisée lors de l'inscription, la découverte d'entités telles qu'une passerelle ou un point d'accès, l'apprentissage d'adresses IP comme celles d'un serveur DHCP ou d'un DNS, la sollicitation de l'EA, etc.

#### 3.2.3.3 Authentification

L'*authentification* est le fondement de la sécurité de toute communication ultérieure car celle-ci ne peut être sécurisée sans que les parties impliquées ne soient assurées de l'authenticité de leurs identités respectives. A cette étape, afin d'accéder au réseau, un client doit s'authentifier en utilisant ses éléments d'accréditation récupérés à l'étape d'inscription. Il doit par exemple montrer qu'il connaît une clef préalablement partagée, qu'il sait chiffrer avec une clef secrète, qu'il sait déchiffrer avec une clef privée, qu'il sait signer une information par sa clef privée, etc. Dans la majorité des cas, dans les réseaux filaires, seule l'authentification du client est requise. Celle du serveur AAA peut être nécessaire dans un réseau sans-fil. On dit alors que l'authentification est *mutuelle*. Divers mécanismes d'authentification existent pour cela.

La phase d'authentification est effectuée par un *mécanisme d'authentification* à distance basé sur un protocole d'authentification. Selon le rôle que joue l'entité à authentifier, qu'elle soit le client ou le serveur, selon la responsabilité qu'elle détient au sein de l'institution gérant le réseau distant, selon la nature du service auquel elle souhaite accéder, le niveau de sécurité nécessaire varie et le protocole d'authentification change en conséquence. Ces facteurs, associés au fait que les

10. S'il s'agit par exemple d'une autorité de certification.

outils cryptographiques évoluent sans cesse, expliquent l'existence d'une grande variété de mécanismes d'authentification.

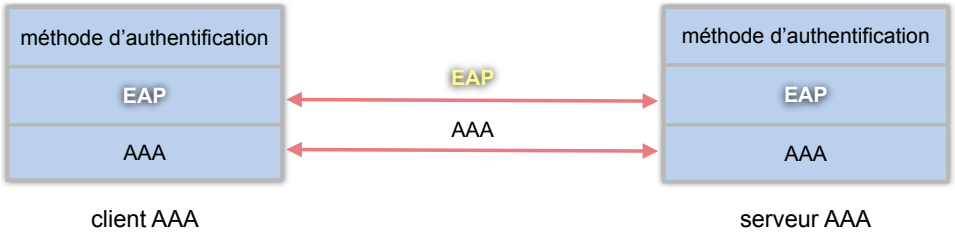


FIGURE 3.4 – Architecture de protocoles proposée

Afin d'éviter un travail de modification d'un protocole AAA chaque fois qu'un nouveau mécanisme d'authentification est conçu, un protocole nommé EAP (*Extensible Authentication Protocol*) a été adapté par l'IETF au contexte AAA. Il définit une nouvelle couche logicielle qui vient s'intercaler entre la couche du protocole AAA et la couche de la méthode d'authentification (cf. FIGURE 3.4). Comme son nom l'indique, il permet, par construction, de transporter toute méthodes d'authentification dont celles non prises en compte ou n'existant tout simplement pas encore lors de sa conception.

D'après le modèle en couche de protocoles (cf. section 2.2.3 du chapitre 2), chaque couche ou sous-couche encapsule [11] ou décapsule [12], selon le sens de traitement, les données reçues de la couche ou sous-couche émettrice. Ainsi, les données de la couche méthode d'authentification sont encapsulées par la couche EAP et le résultat est encapsulé à son tour par la couche AAA.

#### 3.2.3.4 Établissement de clés de session

Une fois l'authentification réussie, le client et l'EA mettent en place des éléments de sécurité communs pouvant servir à chiffrer leur trafic ultérieur, à prouver aussi la légitimité du client et, enfin, à vérifier de façon continue le fait qu'il soit autorisé. Une clé symétrique partagée, appelée souvent *clé de session*, un couple de clefs publique et privée, une clé d'une chaîne de hachage d'un système OTP, une signature numérique d'un ensemble d'information utiles pour la suite des échanges sont autant d'exemple de tels éléments de sécurité. Tout le trafic ultérieur entre le client et l'EA est protégé par les clés de sessions établies.

Plusieurs échanges peuvent avoir lieu pour *négocier* ces paramètres. Cette négociation se fait à l'aide des outils cryptographiques et des protocoles fixés lors de la phase d'initialisation. Il est possible que ces données soient par ailleurs étiquetées par une date d'expiration à l'échéance de laquelle la phase d'établissement d'éléments de sécurité doit être revisitée si le client s'est bien comporté.

11. Encapsuler des données est réalisé par la machine source et consiste à leur ajouter un en-tête sous forme de plusieurs champs contenant des informations destinées à la couche homologue de la machine de destination.

12. Décapsuler des données est effectué par la machine de destination et consiste à enlever l'en-tête qui a été ajouté par la couche homologue de la machine source.

#### 3.2.3.5 Autorisation

Une fois l'identité du client contrôlée et ses clés de sessions ont été générées, ses droits d'accès sont envoyés par le serveur AAA à l'EA afin qu'ils soient mis en application (à vérifier). Parfois le client peut ne pas avoir accès à une ressource bien qu'il y soit autorisé parce qu'elle est simplement indisponible. A l'étape d'autorisation, si le client n'a pas d'adresse IP, il en reçoit une qui lui sera utile pendant toute la durée de sa session.

Quand toutes les phases précédentes ont été exécutées avec succès, le client est autorisée à accéder aux services demandés, protégés par l'EA, et auxquels il a droit.

#### 3.2.3.6 Session autorisée

C'est durant cette étape que le client utilise les services qu'il a souscrits. Ses consommations sont enregistrées par l'EA et périodiquement envoyées au AAAH qui les garde pour établir plus tard une facture. Notons que cela correspond à un modèle de vente post-payée. D'autres modèles commerciaux existent comme le modèle pré-payé.

Par ailleurs, en cas de départ, l'EA note cet événement. Il contrôle aussi la validité des clés de sessions et demande leur renouvellement s'il détecte leur corruption ou expiration. Une surveillance du comportement du client peut être aussi réalisée. Une *traçabilité* ou *traçage* (en anglais *tracing*) permet de vérifier, par ailleurs, si les règles d'accès ont bien été appliquées d'une part, et si elles n'ont pas été violées d'autre part. Pour cela une politique d'enregistrement et de gestion des événements d'accès a été définie. Une alerte à destination de l'administrateur du réseau peut être programmée pour se déclencher quand certains événements se produisent. Ainsi certaines attaques peuvent être détectées et déjouées à temps. Le client peut être, dans ce cas, isolé et ses éléments d'accréditation révoqués. On voit donc que la traçabilité permet de renforcer la sécurité dont une partie est réalisée déjà par l'authentification et les échanges chiffrés entre le client et l'EA.

#### 3.2.3.7 Révocation / Isolation

Ce processus est déclenché si le client adopte un mauvais comportement. La révocation peut consister à mettre certains de ses éléments d'accréditation sur une liste. Par exemple son certificat X.509 peut être ajouté dans une liste de certificats révoqués (en anglais *Certificate Revocation List* ou CRL). Le client ne peut plus s'en servir dorénavant. L'EA peut aussi procédé au filtrage d'une partie ou de la totalité du trafic du client.

## 3.3 Taxonomie

Les caractéristiques des MANETs passées en revue au deuxième chapitre ont montré que leur contexte est principalement distribué et ne repose pas sur une infrastructure d'une ou de plusieurs administrations centralisées et de confiance. Leur exploitation ne peut donc suivre les mêmes schémas que ceux des réseaux à infrastructure. En particulier, la mise en place de services AAA ne peut être immédiate par l'application telle quelle des procédés présentés à la section 3.2.

Plusieurs travaux de recherche se sont penchés sur ce problème. Dans cette section, nous proposons de les classer selon divers critères, d'abord en se référant au degré de dépendance qui lit un MANET à un réseau d'opérateur, ensuite selon l'architecture de services AAA adoptée et les protocoles qui s'y exécutent, puis selon les mécanismes de comptabilisation et de facturation des ressources consommées qui ont été adressés, enfin selon les modes de communication existants entre les composants de l'architecture étudiée (cf. tableau 3.1).

### 3.3.1 Modèle de dépendance

Selon le type d'application, il est possible de définir deux catégories principales de réseaux ad hoc : les réseaux tributaires et les réseaux autonomes. Nous nous référons aux réseaux ad hoc tributaires lorsqu'une étape d'inscription, chez un ou plusieurs FAIs, est nécessaire.Le terme réseau ad hoc autonome désignent, d'autre part, un réseau qui ne nécessite pas d'inscription préalable de ses nœuds.

#### 3.3.1.1 MANET tributaire

C'est évidemment le modèle rencontré dans la mise en œuvre des MANETs pour les exploitations militaires et de secours. Le tiers de confiance est une autorité unique et commune à tous les nœuds du réseau qui partagent ainsi une relation de confiance avec celle-ci. Les nœuds ad hoc partagent, en plus, une relation de confiance entre eux à travers cette autorité.

Par ailleurs, ce modèle s'applique aussi aux exploitations commerciales des MANETs où les opérateurs et les fournisseurs de services se trouvent en position de tiers de confiance veillant à contrôler l'accès aux services qu'ils proposent et à sécuriser leurs ventes. En revanche, pour jouer le rôle de tiers de confiance, un opérateur contracte des accords d'itinérance avec d'autres opérateurs afin de servir ces clients quand ils sont mobiles (cf. section 3.2.1). Ainsi le tiers de confiance n'est pas nécessairement une seule autorité mais une agglomération d'autorités.

Nous divisons cette catégorie de MANETs en deux catégories : les MANETs totalement tributaires où la phase d'inscription des nœuds est indispensable et la présence d'un tiers de confiance pendant les phases ultérieures du processus AAA est nécessaire et les MANETs partiellement tributaires où seule l'inscription est nécessaire mais la présence du tiers de confiance n'est pas requise durant les phases ultérieures sauf, parfois, à la phase de session autorisée pour récupérer des données de comptabilité.

**MANET totalement tributaire** Afin d'illustrer le rôle du tiers de confiance pendant la phase d'authentification, nous nous référons à l'exemple de la figure 3.5. Les opérateurs des domaines $DM_1$ et $DM_2$, notés respectivement $OMS_1$ et $OMS_2$, ont des accords d'itinérance avec l'opérateur FAI DV. Les nœuds mobiles $MN_1$ et $MN_2$ sont deux clients abonnés respectivement à $OMS_1$ et $OMS_2$. L'authentification de $MN_1$ (respectivement de $MN_2$) est effectuée par le serveur AAAH du $DM_1$ (respectivement par le serveur AAAH du $DM_2$) à travers DV par l'intermédiaire de AAAV. Une partie de cette signalisation est tracée en guise de démonstration.

Suite à cela, DV autorise $MN_1$ (respectivement $MN_2$) à accéder aux services souscrits. Dorénavant $MN_1$ (respectivement $MN_2$) et DV partagent une relation

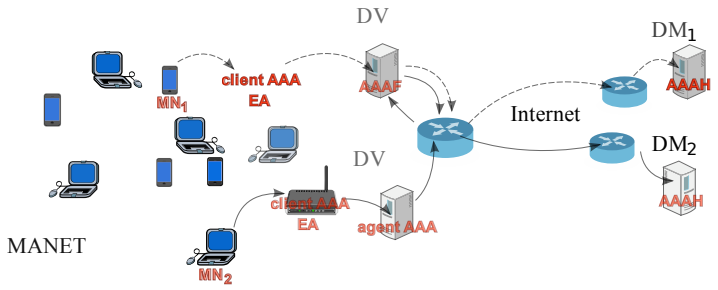


FIGURE 3.5 – MANET tributaire

de confiance tant que $MN_1$ (respectivement $MN_2$) est autorisé. Régulièrement, un compte rendu des consommations de ces deux nœuds est envoyé par le AAAV au AAAH de leur OMSs respectifs. Si $MN_1$ et $MN_2$ souhaitent établir une relation de confiance, ils peuvent utiliser DV comme tiers de confiance pour mettre en place une clé partagée (cf. section 2.5.1 du chapitre 2).

La solution décrite par Sargento et al. dans [61] dans le cadre du projet européen DAIDALOS obéit à ce schéma. Celle présentée par Chaouchi et Laurent dans [62] suit ce schéma mais au lieu d'avoir deux OMSs, $OMS_1$ et $OMS_2$, un seul OMS a été considéré. De plus, des nœuds ad hoc particuliers secondent l'architecture AAA centrale et offrent un ou plusieurs services parmi les trois services AAA. De même, Kounga et Schaefer supposent dans [63], travail réalisé à DoCoMo Euro-Labs, l'existence d'une autorité mère sous forme d'un fournisseur de services de contenus.

Moustafa et al. dans [64] font appel à un serveur AAA central employant Kerberos pour fournir des services AAA dans MANET. Mais tout nœud ad hoc est aussi capable d'authentifier un nœud voisin avant de lui confier le relayage de son trafic. Au sein du projet finlandais SESSI, Leggio et al. dans [65] supposent l'existence d'un service AAA central dans le réseau d'un opérateur à proximité du réseau MANET sans pour autant spécifier son architecture. Enfin, Zhong et al. [66] font l'hypothèse de l'existence d'un serveur central pour les opérations de facturation qu'ils désignent par l'appellation *Service de compensation* (en anglais *Credit Clearance Service* ou CCS). Leur solution ne traite pas des aspects AAA au complet, notamment des services d'authentification et d'autorisation. Mais elle est intéressante à examiner dans la mesure où elle étudie un modèle de facturation ambitionnant d'annuler le problème des nœuds égoïstes (cf. tableau 3.1).

**MANET partiellement tributaire** Tous les nœuds ad hoc ou une partie d'entre eux sont capables d'opérer comme serveurs AAA. Cela dépend de l'architecture choisie. Nous examinerons ce point à la section 3.3.2. De ce fait, à part pour l'initialisation des éléments d'accréditation et/ou l'envoi des données de comptabilité, le tiers de confiance n'est pas indispensable au fonctionnement du réseau. Les serveurs AAA, se trouvant au sein du MANET, traitent le reste des étapes du processus AAA.

Lamsal était parmi les premiers à avoir proposé une solution de ce type [67] [68]. WATCHMAN [69] s'intègre aussi dans cette catégorie ne dépendant que

partiellement d'un opérateur (cf. tableau 3.1).

#### 3.3.1.2 MANET autonome

A notre connaissance, il n'existe pas de solutions AAA pour les MANETs autonomes (cf. tableau 3.1). Les enjeux financiers ne supportant pas l'absence d'un tiers de confiance responsable de la mise en place de relations sûres, cette situation est plutôt logique. Toutefois, il est possible d'imaginer des services combinant l'authentification, l'autorisation, et la traçabilité, que nous pouvons appeler AAT, ayant pour but simplement de contrôler l'accès à certains services non payants comme ceux présentés à la section 1.3.3.2.

### 3.3.2 Modèle d'architecture

Nous divisons les architectures rencontrées dans la littérature en trois catégories : centrale, hiérarchique centrale, hiérarchique distribuée et pair-à-pair. Toutes ont été conçues pour des MANETs de type tributaire.

Afin de pouvoir les présenter, nous adopterons les notations suivantes :

- $\mathbf{A}_1$ pour désigner une entité réalisant un service d'**authentification**
- $\mathbf{A}_2$ pour désigner une entité réalisant un service d'**autorisation**
- $\mathbf{A}_3$ pour désigner une entité réalisant un service de **comptabilité**

Ainsi une entité ne pouvant réaliser que des tâches d'authentification et d'autorisation est notée $A_1A_2$, une deuxième ne pouvant réaliser que des tâches d'authentification et de comptabilité est notée $A_1A_3$, enfin, une troisième ne pouvant effectuer que des tâches d'autorisation et de comptabilité est notée $A_2A_3$. La tâche de comptabilité peut par exemple consister à compter le nombre de paquets ou à mesurer le volume de données consommé par d'autres entités et à envoyer régulièrement un rapport de ces consommations à un serveur AAA central.

Nous verrons, d'autre part, qu'au sein des MANETs utilisant ces architectures, un nœud ad hoc passe une suite d'états dont un aperçu est donné à la figure 3.6. Avant son abonnement à un OMS, un nœud ad hoc, MN se trouve dans un état de nœud inconnu. Après inscription et initialisation (cf. FIGURE 3.3), il devient nœud abonné. Tant qu'il n'a pas à sa portée un réseau ad hoc en opération ou que son terminal associé est éteint, il reste absent de tout réseau MANET. Quand il arrive à proximité d'un MANET et que son terminal est allumé, il passe à l'état de nœud arrivant. Plusieurs mécanismes se mettent alors en route afin, par exemple, de découvrir les nœuds qui sont dans son voisinage immédiat [70], pour configurer une adresse IP [71] à laquelle il sera joignable et de laquelle il enverra du trafic, etc. Mais il ne pourra pas bénéficier des services offerts tant qu'il n'a pas été authentifié par le réseau. Pour cette raison, il passe à l'état de nœud en cours d'authentification.

L'authentification du MN est prise en charge par les différentes entités du réseau MANET et/ou du réseau de l'opérateur qui sont autorisées à le faire. Si l'authentification réussit, il passe à l'état de nœud en cours d'autorisation en acquérant une ou plusieurs clés de sessions. Si son autorisation à utiliser les services auxquels il a droit est établie, il passe à l'état de nœud autorisé et il y reste tant que ses clés de session n'ont pas expiré, qu'il n'a pas été identifié comme nœud malveillant suite à un mauvais comportement, ou qu'il n'a pas

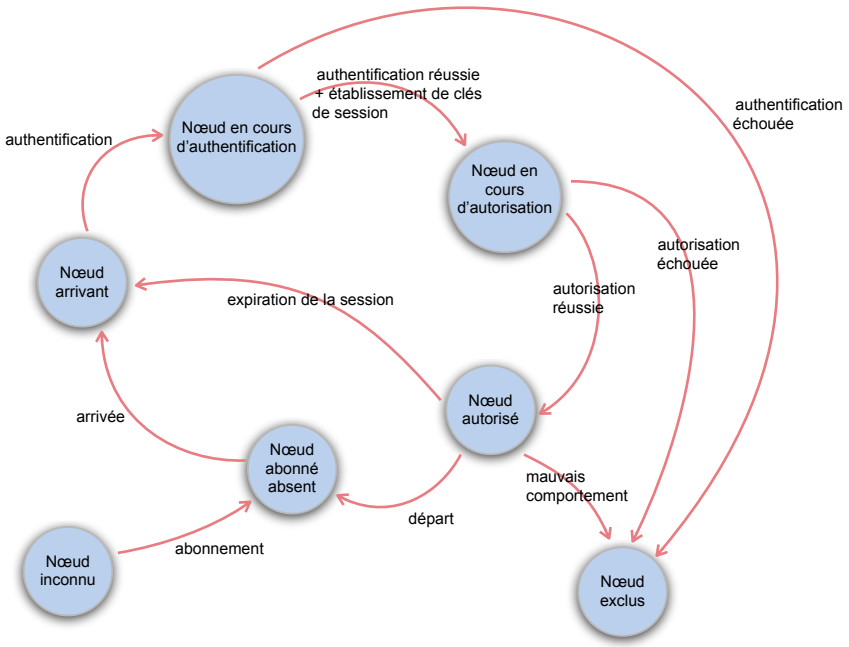


FIGURE 3.6 – Diagramme d'états d'un nœud ad hoc

quitté le réseau. Si ses clés expirent, il doit passer à l'état de nœud arrivant pour une nouvelle authentification.

#### 3.3.2.1 Architecture centrale

C'est une architecture qui emploie un seul serveur AAA central qu'il soit hébergé dans le réseau d'un opérateur dans le cas d'un MANET totalement tributaire ou présent dans le réseau ad hoc lui-même dans le cas d'un MANET partiellement tributaire. De telles architectures sont quasiment inexistantes dans la littérature. Elles peuvent être évoquées comme hypothèse. Nous l'avons constaté dans [66] qui suppose l'existence d'un serveur CCS sans vraiment s'étendre sur les détails de ce serveur et sur son éventuelle appartenance à une architecture AAA hiérarchique centrale.

D'autre part, les articles que nous avons explorés montrent que les chercheurs s'accordent sur l'idée qu'un serveur AAA unique dans un MANET n'est pas une solution viable pour des raisons de sécurité et de sûreté de fonctionnement. Le service AAA doit, en effet, rester opérationnel même si ce serveur quitte le réseau, tombe en panne ou succombe à des attaques en provenance de nœuds malveillants.

#### 3.3.2.2 Architecture hiérarchique centrale

Il s'agit d'une architecture comprenant un serveur AAA central et d'autres entités subordonnées telles que un ou plusieurs serveurs AAA, agents AAA [13] ou

13. En général, quand un nœud ad hoc est serveur ou proxy AAA, il offre en plus d'autres services comme un service de mobilité [61], un service d'adressage [62], etc. L'hypothèse men-

entités $A_1$, $A_2$, $A_3$, $A_1A_2$, $A_1A_3$, ou $A_2A_3$. Un nœud ad hoc peut se voir confier le rôle d'entité subordonnée par son opérateur. Mais il ne pourra l'exercer que quand il est dans l'état de nœud autorisé (cf. FIGURE 3.6). Quoiqu'il en soit, il doit toujours jouer au moins le rôle de client.

Le modèle d'architecture hiérarchique centrale est le modèle dominant dans la littérature. On le rencontre souvent dans le cas des MANETs totalement tributaires où des nœuds ad hoc offrent certains services AAA complétant ceux offerts par l'opérateur et étendant ainsi l'architecture AAA hiérarchique centrale de celui-ci.

Par exemple, Chaouchi et Laurent [62] supposent que certains nœuds ad hoc sont *homologués* par l'opérateur afin de leur conférer un rôle de $A_1$, de $A_2$, de $A_3$, de $A_1A_2$ ou de $A_2A_3$ dans le MANET. Précisons au passage que les auteurs supposent aussi que l'EA (cf. FIGURE 3.5) héberge un proxy AAA ou un serveur AAA à la place d'un client AAA comme nous l'avons présenté précédemment (cf. section 3.2.1). Par exemple encore, dans la solution de Moustafa et al. [64], chaque nœud ad hoc est $A_1A_2$ pour chaque nœud voisin. Comme dans la solution précédente, dans l'EA, il existe un proxy Kerberos, mandataire du serveur Kerberos de l'opérateur.

Dans DAIDALOS [61], quoique les nœuds ad hoc ne supportent pas les fonctions d'authentification et d'autorisation, laissées intégralement au serveur AAA de l'opérateur[14] (cf. section 3.2.1), ils assurent conjointement la tâche de comptabilité. Ils sont donc nœuds $A_3$. Pour notifier le serveur AAA des consommations des uns et des autres, ils suivent la solution proposée dans [66].

En ce qui concerne le projet SESSI, chaque nœud abrite un module d'authentification et d'autorisation (nœuds $A_1A_2$) dans le cadre d'une architecture SIP (*Session Initiation Protocol*) distribuée. Enfin, pour ce qui est de la solution étudiée par DoCoMo [63], certains nœuds du MANET sont distributeurs de contenus pour le compte d'un fournisseur de contenus ayant contracté des accords d'itinérance avec un opérateur d'accès. Bien que les auteurs n'aient pas précisé si un nœud distributeur était un serveur AAA, celui-ci est capable d'effectuer les tâches AAA lors d'une opération de vente avec un nœud ad hoc consommateur. Néanmoins, en matière d'authentification et de comptabilité, il s'appuie en partie sur le serveur AAA de l'opérateur.

Dans la catégorie des MANETs partiellement tributaires, le modèle d'architecture centrale hiérarchique est parfois utilisé. Un serveur central dans le réseau MANET lui-même est pour cela élu ou désigné par un opérateur. D'autres serveurs secondaires, parfois appelés esclaves, ou entités $A_1$, $A_2$, $A_3$, $A_1A_2$, $A_1A_3$, $A_2A_3$ l'assistent, par ailleurs, dans ses fonctions.

Le serveur principal et le serveur secondaire de Lamsal [68] sont élus par les clients selon un critère lié à leur durée de connexion au réseau : le serveur principal est le nœud ad hoc ayant la durée de connexion la plus longue suivi par le serveur secondaire. Préalablement à tout départ, le serveur principal prévient le serveur secondaire afin que celui-ci puisse prendre la relève et diriger les élections d'un nouveau serveur secondaire. De même, avant de partir, le serveur

tionnant qu'un tel nœud doit être à mobilité réduite est souvent employée. Ainsi son déplacement reste cantonné au voisinage immédiat d'une même passerelle réseau ou de l'EA (cf. FIGURE 3.5).

14. Ces flux passent à travers le MANET avant d'arriver à l'EA, ce qui risque d'engendrer des problèmes de sécurité car ils étaient au départ conçus pour atteindre l'EA en un seul saut et non en plusieurs.

secondaire prévient le serveur principal qui organise alors les élections d'un nouveau serveur secondaire. Si les deux serveurs arrivent à disparaître au même moment, un client se proclame serveur temporaire et organise les élections d'un nouveau serveur principal qui se charge plus tard des élections d'un nouveau serveur secondaire. Un échange fréquent entre les deux serveurs d'une part et la diffusion par chacun d'entre eux d'un message à tous les clients d'autre part permet de détecter leur éventuel départ ou leur disparition.

Le serveur maître (*Master AAA server* ou MAS) dans un MANET de WATCHMAN[69] est soit choisi par l'opérateur soit élu par d'autres nœuds serveurs AAA appelés esclaves (*Slave AAA Servers* ou SASs). Les SASs sont élus par le MAS en se basant sur la valeur de la quantité suivante :

$$w = A.\frac{Rc\,R + Tc\,T + Lc\,L}{3}$$

où le facteur $A \in \{0, 1\}$ (*AAA server node factor*) exprime la volonté d'un nœud de devenir serveur ou pas, le paramètre de ressources $R$ (*Resources*) reflète les capacités physiques du nœud en terme de puissance de calculs, d'espace mémoire, etc., le paramètre de confiance $T$ (*Authorization Trust Level*) exprime le niveau de confiance assigné par l'opérateur au nœud considéré, le paramètre $L$ (*Location*) reflète la position du nœud par rapport au MAS et les coefficients $Rc$, $Tc$, $Lc$ sont fixés par l'opérateur selon l'importance qu'il accorde à chacun de ces trois paramètres. Plus $w$ est grand, meilleures sont les chances d'un nœud pour devenir SAS.

Grâce à l'utilisation d'un minuteur, il est possible de détecter l'indisponibilité du MAS. Le SAS ayant le SAS le plus élevé se proclame alors MAS. D'autre part, le MAS est en mesure de relever un SAS de ces fonctions s'il détecte la présence d'un nœud ayant une meilleure valeur de $w$.

#### 3.3.2.3 Architecture hiérarchique distribuée

Dans cette architecture, plusieurs serveurs AAA offrent les trois services et aucun d'entre eux ne centralise les opérations. Lamsal, dans le même article [68], a proposé une solution de ce type qu'il a appelée *Modèle du serveur de groupe* où tout nœud ad hoc devient momentanément serveur AAA lorsqu'il est sollicité par un nœud client arrivant dans le MANET et demandant une autorisation d'accès au réseau. Ce nœud serveur consulte alors un groupe de nœuds ad hoc, supportant chacun des fonctions de types AAA, afin de décider collectivement de l'admission ou non admission du nœud arrivant. Le nombre de nœuds sollicités peut être le nombre total des nœuds du réseau ou un nombre seuil d'entre eux noté $m$. Quel que soit ce nombre, seulement $m$ réponses sont prises en compte dans le processus de décision. Lamsal pense que la cryptographie à seuil pourrait être un moyen pour accomplir ce processus de décision collective.

#### 3.3.2.4 Architecture pair-à-pair

Tous les nœuds sont clients et serveurs AAA à la fois. Il s'agit donc de pairs. Quand un nœud mobile souhaite appartenir au réseau, il agit comme un client et demande l'accès à un nœud voisin faisant déjà partie du réseau et qui est par conséquent serveur AAA [68].

### 3.3.3 Services commercialisables et comptabilité

Les solutions ayant traités des services AAA se classent toutes dans la catégorie des MANETs tributaires. Elles adoptent majoritairement un système de comptabilité reposant sur un ou plusieurs serveurs AAA centraux quoique les comptabilisations des consommations soient souvent réalisées par des nœuds ad hoc individuels. Ceux-ci rendent ensuite compte de cette comptabilité aux serveurs AAA.

#### 3.3.3.1 Types de services

Nous distinguons quatre sortes de services : les services accédés à travers Internet, les services d'accès à Internet, les services de contenus et les services d'accès au MANET.

On a vu au chapitre 1 à la section 1.3.4 que plusieurs services peuvent être fournis aux nœuds d'un MANET lorsque celui-ci est connecté à Internet. Nous désignons ces services par *services à travers Internet*. Afin de pouvoir les acheminer, il est important qu'un ou plusieurs opérateurs de souscription offrent des *services d'accès à Internet*, que ce soit directement ou par itinérance, à certains ou tous les nœuds du MANET. Les services d'accès à Internet consistent essentiellement à attribuer une adresse à un nœud ad hoc pour qu'il puisse devenir joignable (il s'agit du *service d'adressage*), à définir sa bande passante, à router son trafic à travers Internet (c'est le *service de routage*) et à assurer le rattachement du client à Internet quand celui-ci se déplace (c'est le *service de mobilité*). Mais, les contrats portant sur les services d'accès à Internet comportent souvent d'autres services supplémentaires, comme l'hébergement de sites web ou l'accès à une boîte électronique, etc. Nous les rangeons dans la catégorie des services à travers Internet.

D'autre part, des nœuds ad hoc *distributeurs* peuvent vendre des contenus multimédia ou des renseignements pour le compte de fournisseurs auprès desquels ils sont mandatés. On parlera alors de *services de contenus*. Les nœuds acheteurs ainsi que les nœuds vendeurs doivent avoir conclu un contrat avec leurs opérateurs lors de leurs souscriptions afin de pouvoir exercer ces rôles [63]. Selon que le service demandé est un service d'accès à Internet ou un service de contenus, le déroulement des opérations de commercialisation diffère et les protocoles impliqués diffèrent aussi (cf. section 3.3.3.2).

Quoiqu'il en soit, il n'est pas possible, en pratique, de rendre ces services sans fournir certains autres services dans le MANET lui-même tels que l'adressage afin que les nœuds ad hoc n'ayant pas souscrit aux services d'un opérateur ou ne pouvant pas joindre directement leur opérateur puissent avoir une adresse. Ce cas peut arriver notamment quand un nœud est dans l'état de nœud arrivant (cf. FIGURE 3.6) et qu'il n'est pas à la portée d'une passerelle d'un opérateur. Il doit pouvoir avoir une adresse, même provisoire, pour émettre du trafic comme celui nécessaire à son authentification. Un autre service important à mentionner est le routage à travers le réseau MANET qui est, rappelons le, fourni par tout nœud ad hoc se trouvant en position de relais sur un chemin donné (cf. section 1.2.1 du chapitre 1). Enfin, un service non moins utile est celui d'annonce de la disponibilité et de l'emplacement de certains de ces services. Nous désignons l'ensemble de ces services par *services d'accès au MANET*.

En outre, les services AAA assurés, selon le type d'architecture (cf. section

3.3.2), par l'opérateur, conjointement par l'opérateur et par le MANET, ou par le MANET seulement doivent aussi être pris en compte dans la liste des services fournis aux nœuds ad hoc.

Dans [62], les services décrits ci-dessus sont tous disponibles et payants. Les nœuds homologués offrent un service de contenus, d'authentification, d'autorisation, de comptabilité et d'adressage temporaire aux nœuds arrivant en attendant qu'ils soient authentifiés avec succès par le serveur AAA de l'opérateur qui se charge alors de leur attribuer des adresses définitives.

Les solutions WATCHMAN [69] et SESSI [65] concernent les services d'accès à Internet et les services accédés à travers Internet. DAIDALOS s'intéresse en plus aux services d'accès à un MANET [61] et Moustafa et al. [64] comme DoCoMo [63] aux services de contenus. Sprite [66], en revanche, s'est concentré sur l'étude de comptabilité et de facturation du service de routage dans un MANET (cf. section 3.3.3.2).

#### 3.3.3.2 Méthodes de comptabilisation

Chacune des catégories de services décrites ci-dessus donne lieu à un mode de comptabilisation différent. La comptabilité des services d'accès à Internet et des services à travers Internet peut suivre une procédure exécutée habituellement par l'architecture AAA du ou des opérateurs. Celle des services d'accès au MANET et des services de contenus doit, quant à elle, être différente puisque les nœuds ad hoc sont aussi bien consommateurs que fournisseurs d'une partie ou de la totalité de ces services. Nous présenterons donc les aspects liés à la comptabilité de ces derniers, laissant ainsi de côté les modes de comptabilité en rapport avec les deux premiers types de services.

**Services d'accès au MANET : routage** La comptabilité des services d'accès au MANET repose principalement sur l'observation du trafic des nœuds consommateurs, c.-à-d. du trafic routé. En effet, puisque les services d'adressage et d'annonce sont de toute manière fournis à tout nœud ad hoc, seul le service de routage compte réellement dans les opérations de comptabilité. Ainsi il est possible pour un nœud relais de compter le nombre de paquets envoyés par une source et qu'il a relayés ou d'additionner leur volume. Il lui est aussi possible d'enregistrer les dates de début et de fin d'émission de la source [69]. Il envoie, par la suite, un compte rendu de ces consommations au serveur AAA central. Cet envoi peut avoir lieu de façon périodique, après une période d'inactivité de la source ou après détection de son départ ou de son déplacement [62].

**Services de contenus** La comptabilité des services de contenus, tout comme celle du service de routage dans un MANET, peut reposer sur le comptage du nombre de paquets de *contenus* reçu par un consommateur, ou sur leur volume ou encore sur la durée d'utilisation du service de contenus [62].

**Facturation** La facturation peut être postérieure aux consommations, on parle alors de mode de facturation à la consommation réelle, ou antérieure et on parle alors de mode de facturation pré-payée. Elle repose sur les éléments de comptabilité tels que décrits dans les sections 3.3.3.2 et 3.3.3.2, mais aussi sur le nombre de services dont un client peut ou a bénéficié. Rappelons que les

services AAA eux-même peuvent être facturés avec le reste des services évoqués plus haut (cf. section 3.3.3.1). De plus, l'activation d'un service donné peut être unique en mode pré-payé [15] ou multiple [16] quel que soit le mode [62].

La facturation de l'ensemble des services fournis à travers un MANET peut être décidée de telle sorte qu'elle avantage un consommateur se trouvant dans un MANET à celui directement attaché à une passerelle d'un opérateur. Ceci est envisageable car les nœuds ad hoc sont non seulement consommateurs mais aussi au moins fournisseur de service de routage voire de service de contenus.

Ainsi un nœud ayant acheminé un trafic peut être rémunéré et la source du trafic [62] ou sa destination facturée [66]. C'est la méthode par débit et par crédit (en anglais *Charging and rewarding*). Son objectif principal est de remédier au problème de nœuds égoïstes en les encourageant de cette manière à participer aux opérations de routage [17]. Elle a été d'abord développée dans le cadre du projet Terminodes où Buttyàn et Hubaux [72] ont proposé deux modèles, pour simuler la coopération entre nœuds ad hoc, se servant d'une monnaie virtuelle appelée *nuglet*. Dans le premier modèle, *Packet Purse model*, une source charge un paquet, avant de l'envoyer, avec un nombre suffisant de *nuglets* afin qu'il atteigne la destination. Chaque nœud intermédiaire prend alors des *nuglets* en guise de rémunération pour son service de relayage. Dans le deuxième modèle, *Packet Trade Model*, chaque nœud relais achète le paquets pour quelques *nuglets* et le vend au nœud relais suivant contre plus de *nugets* jusqu'à l'arrivée à la destination qui l'achète. La mise en œuvre de ces systèmes nécessite que l'équipement de chaque nœud soit inviolable afin d'éviter tout acte de fraude.

Dans Sprite [66], Zhong et al. ont repris cette idée mais en faisant appel à un CCS se passant ainsi d'équipements inviolables. Chaque nœud relais garde un reçu du paquet qu'il a routé et envoie régulièrement l'ensemble des reçus au CCS dans le but que celui-ci le rémunère. D'autre part, le CSS débite une source qui a émis un paquet en fonction des reçus recueillis pour ce paquet, du nombre de nœud relais restant à traverser jusqu'à l'arrivée à la destination et de la réception ou non du reçu de celle-ci. La conception du système est telle que les nœuds n'ont pas intérêt à mal agir essentiellement pour des raisons économiques.

15. Le nœud ad hoc consommateur ne peut activer le service qu'une seule fois et il n'a pas la possibilité de le suspendre.

16. Il est possible de suspendre le service et d'avoir donc des périodes de suspension contrairement au mode pré-payé à activation unique.

17. Il existe d'autres solutions au problème de nœuds égoïstes, notamment les systèmes basés sur la réputation, les systèmes basés sur l'utilisation d'un *watchdog* et de notation des chemins. Dans les systèmes basés sur la réputation, chaque nœud surveille ses nœuds voisins afin de déterminer le niveau de confiance qu'il peut leur accorder ou s'il faut, au contraire, les exclure de son cercle de confiance. Dans les systèmes *watchdog*, un nœud relais $R_i$ garde une copie du message qu'il a transmis à son voisin $R_{i+1}$, se trouvant sur le chemin entre la source $S$ du message et la destination $D$, et surveille si $R_{i+1}$ l'a relayé correctement. Si cela n'est pas le cas, il augmente sa note de défaillance. Quand celle-ci dépasse un seuil défini, $R_i$ informe $S$ que $R_{i+1}$ est un nœud malveillant. De cette manière, il est possible pour un nœud de calculer une note globale pour chaque chemin amenant à une destination dans le but de choisir le ou les chemins les plus sûrs.

### 3.3.4 Modes de communication AAA

La majorité des protocoles existants dans les solutions que l'on a recensées sont du type *one-to-one*[18] qu'il s'agisse de protocoles d'authentification, d'autorisation ou de comptabilité. Mais Lamsal a émis l'idée de protocoles *one-to many*[19] en matière d'authentification. Nous examinons ci-dessous ces deux catégories.

#### 3.3.4.1 Protocoles one-to-one

Ce type de protocole a lieu entre un nœud ad hoc client et un serveur AAA, à travers éventuellement des entités subordonnées (cf. section 3.3.2.2), ou entre deux nœuds ad hoc l'un client et l'autre une entité subordonnée mais qui n'est pas un agent AAA comme $A_1$, $A_2$, $A_3$, $A_1A_2$, $A_1A_3$ ou $A_2A_3$ (cf. paragraphe 3.2.1). Il est utilisé dans les architectures AAA hiérarchiques centrales que ce soit celles des MANETs totalement tributaires ou celles des MANETs partiellement tributaires.

**MANET totalement tributaire à architecture AAA hiérarchique centrale** La figure 3.7 représente un diagramme de séquences du processus AAA dans ce type de MANET et récapitule l'ensemble des protocoles one-to-one que l'on peut rencontrer dans la littérature. MN est un client souhaitant rejoindre le réseau pour bénéficier des services qui y sont fournis et participer à ces différentes opérations, voire offrir un service comme un service de routage ou un service de contenus. $R_1$ est son nœud voisin relayant son trafic, D est un nœud distributeur de contenus et $R_2$ est un nœud se trouvant à la portée de la passerelle GW du réseau du FAI du DV. $R_1$ tout comme D et $R_2$ ont déjà été authentifiés et autorisés pour pouvoir jouer leurs rôles respectifs auprès du MN, tandis que ce dernier cherche à s'authentifier et à être autorisé.

Sachant que les nœuds ad hoc sont abonnés à un ou plusieurs OMS, l'étape d'initialisation se déroule de la même manière que dans les réseaux à infrastructure c.-à-d. comme évoqué au paragraphe 3.2.3.1 de la section 3.2.3. Les cas particuliers de fin après départ, de révocation ou d'isolation en cas de mauvais comportement et de ré-authentification en cas d'expiration de la session n'y sont pas représentés. Lorsque MN arrive à la porté de $R_1$ appartenant au MANET, avant toute authentification, son adresse IP doit être configurée, suite à quoi il commence à recevoir les annonces des services disponibles.

Plusieurs méthodes de configuration d'adresse IP existent, d'abord l'auto-configuration d'adresse [73] [71], ensuite l'attribution d'une adresse provisoire par $R_1$ au MN, comme suggéré par [62], enfin, la configuration par le MN d'une adresse IP générée de manière cryptographique (*Cryptographically Genenerated Address* ou CGA [74]) après récupération du préfixe du réseau de la part du nœud $R_2$, comme montré dans [61]. CGA nécessite habituellement l'exécution

18. Un protocole one-to-one est un mode de communication qui implique uniquement deux entités : une première entité qui initie les échanges et une deuxième qui lui répond. Une communication entre un client et un serveur est un exemple d'une telle communication.

19. Un protocole one-to-many est un mode de communication qui implique une entité unique initiatrice des échanges et plusieurs entités réceptrices qui lui répondent. Une communication RTS/CTS (cf. section 1.4.1) est un exemple d'une telle communication : le nœud souhaitant réserver le canal envoie un RTS à tous ses voisins qui lui répondent, chacun, quand cela est possible, par un message CTS.

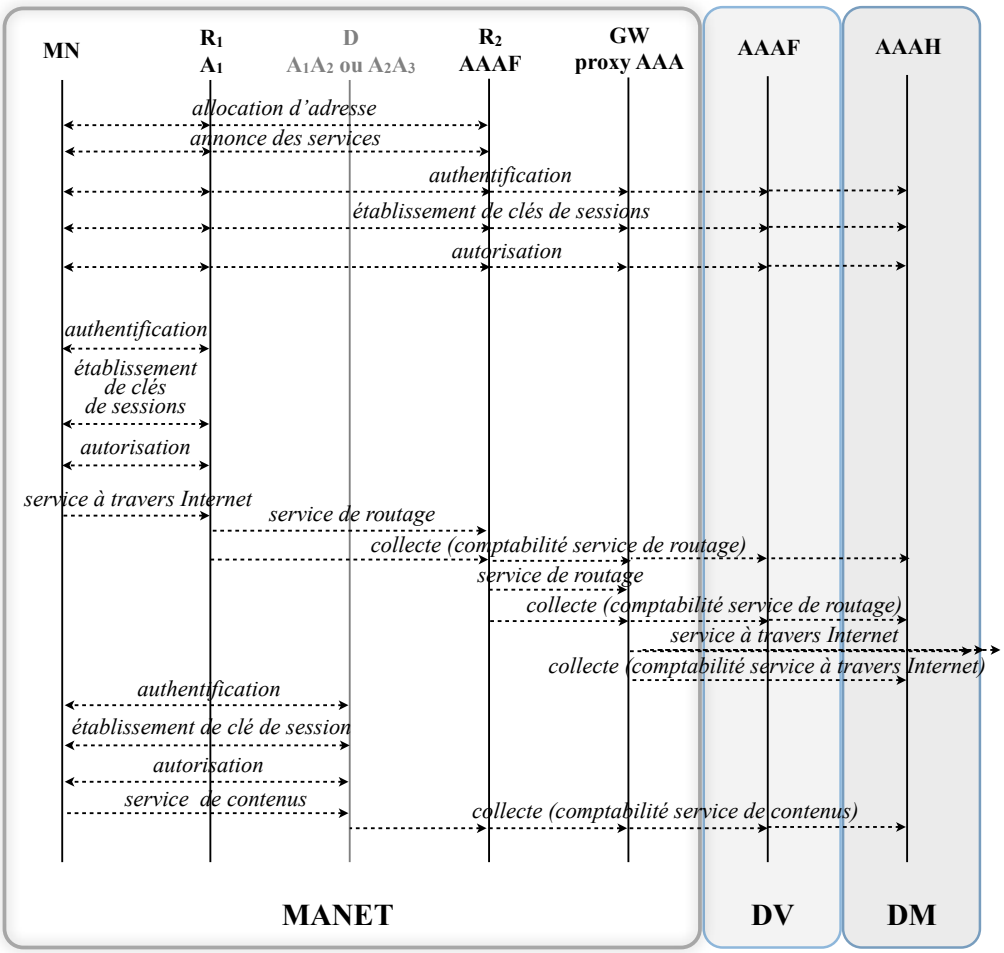


FIGURE 3.7 – MANET totalement tributaire à architecture AAA hiérarchique centrale : diagramme de séquences

du protocole SEND (*SEcure Neighbor Discovery*) [70] entre nœuds voisins, en l'occurrence ici entre MN et son voisin $R_1$. Sargento et al. [61] propose d'étendre ce fonctionnement au delà du voisinage c.-à-d. entre MN et $R_2$. Malheureusement les auteurs n'ont pas détaillé ce fonctionnement.

Sur la figure 3.7, nous remarquons que l'on retrouve les étapes du processus AAA classique (cf. section 3.2.3) à savoir l'authentification, l'établissement de clés de session, l'autorisation et la session autorisée où la comptabilité est effectuée. Néanmoins, vu le contexte distribué d'un MANET et l'absence d'une entité centralisant les opérations du réseau, les trois étapes d'authentification, d'établissement de clés de sessions et d'autorisation ont lieu à trois niveaux distincts : entre le client MN et le AAAH, entre le MN et son voisin relais $R_1$, entre le MN et D. La première fois, l'authentification sert à établir une première relation de confiance basée sur les éléments d'accréditation du MN acquis à l'étape d'initialisation. L'établissement de clés de sessions et l'autorisation servent à permettre de faire valoir ses droits futurs vis-à-vis des divers acteurs : réseau du

FAI du DV, nœuds relais et nœuds distributeurs. La deuxième fois, elles servent à établir une relation de confiance dans le voisinage afin que le $R_1$ soit assuré de l'identité du MN et de son droit au routage. Enfin, la troisième fois, elles servent à établir une relation de confiance entre le MN et D afin que celui-ci soit assuré de l'identité du MN et de ses droits d'utilisation des services de contenus qu'il propose. Notons que ce procédé suit en cela le modèle en couches d'oignon où chacun des trois acteurs évoqués présente un domaine distinct (cf. section 2.3.2 du chapitre 2).

Remarquons, par ailleurs, qu'alors que la première authentification est toujours présente dans la littérature, une simple autorisation peut être suffisante à la deuxième et troisième fois. Cela dépend en réalité de la manière dont la relation de confiance est définie :

- l'authentification par un serveur central seul suffit pour qu' une relation de confiance soit établie entre le nœud nouvellement authentifié et tous les nœuds déjà présents dans le réseau. Dans ce cas, il n'y a nul besoin d'une authentification ultérieure.
- l'authentification par un serveur central seul ne suffit pas pour établir une relation de confiance avec certains nœuds comme les nœuds voisins, directement impliqués dans les opérations de routage du trafic du nœud arrivant, ou les nœuds distributeurs quand celui-ci les sollicite.

1. *Protocoles entre MN et AAAH :* nous en avons donné un aperçu à la section 3.3.1.1 et illustré cela à la figure 3.5. Pour effectuer concrètement l'authentification d'un nœud ad hoc client, DAIDALOS [61] propose d'étendre l'utilisation du protocole PANA (*Protocol for carrying Authentication for Network Access*) au contexte multi-saut des MANET en permettant de créer un tunnel entre ce nœud et l'EA ou le routeur d'accès. Ainsi les messages PANA se retrouvent encapsulés dans ce tunnel.

   Moustafa et al. [64] proposent d'étendre le modèle d'authentification de Kerberos grâce à la médiation du proxy qui relaye les demandes d'authentification de chaque nœud ad hoc au serveur Kerberos. Le protocole d'authentification est à deux échanges. Le contenu des messages est le même que celui du protocole Kerberos de base quand ceux-ci circulent entre le proxy et le serveur mais il est modifié quand ils circulent entre le proxy et un nœud ad hoc. Les auteurs ajoutent en effet un drapeau à ces messages afin de distinguer ce protocole d'authentification pour nœuds ad hoc du protocole Kerberos ordinaire. En plus, dans le dernier message, TGS-REP, la structure de données autorisées est modifiée afin de délivrer plusieurs clés symétriques au MN au lieu d'une seule. Ces clés, appelées *Kadhoc-auth*, serviront à des authentifications ultérieures ainsi que nous le décrivons dans les deux points suivants.

   En plus des clés, un ou plusieurs tickets sont aussi envoyés au MN comme éléments d'autorisation. Dans [62], AAAH envoie plutôt un identifiant, un certificat ou un jeton qui permettent au MN de faire valoir ses droits. Une adresse IP définitive est aussi envoyée pour que le MN puisse remplacer son adresse provisoire. Par la suite, $R_1$ et $R_2$ ajoutent une entrée pour le MN dans leurs tables respectives de *nœuds authentifiés* afin de s'y référer lors des prochains routages du trafic du MN. Ils peuvent par ailleurs demander une authentification supplémentaire au MN [64].

2. *Protocole entre MN et $R_1$ :* une clé *Kadhoc-auth* est employée par le pro-

tocole EAP-PSK pour effectuer une authentification mutuelle entre deux nœuds voisins [64].

Habituellement, le système Kerberos utilise une clé symétrique partagée entre le client et le serveur pour les opérations d'authentification. Moustafa et al. [64] proposent de se servir, de la même manière, de la clé partagée entre les nœuds ad hoc afin que deux nœuds quelconques puissent s'authentifier mutuellement. Mais, une clé symétrique est plus vulnérable à la divulgation qu'une clé asymétrique (cf. section 2.5 du chapitre 2). De plus, cette vulnérabilité augmente avec le nombre d'entités connaissant la même clé. Pour cette raison, les auteurs divisent le temps en $N$ intervalles de même durée $d$ telle qu'à chaque intervalle, une seule clé symétrique est valide et doit être utilisée par tous les nœuds [20].

Le FAI fixe un nombre $n \leq N$ de clés à envoyer à chaque nœud par le serveur Kerberos dans le message TGS-REP. L'intersection entre deux ensembles de clés *Kadhoc-auth* appartenant chacun à un nœud ad hoc distinct, dont la session est valide à la date $t$, est nécessairement non vide. Supposons que MN et $R_1$ sont un tel couple de nœuds, alors ils peuvent s'authentifier mutuellement avec $Kadhoc\text{-}auth_t$.

Dorénavant, dès que MN se fera relayer son trafic, les nœuds intermédiaires comme $R_1$ commenceront à en effectuer la comptabilité (cf. section 3.3.3.2). Par ailleurs, les services d'accès à Internet et par Internet consommés par le MN sont aussi comptabilisés par le AAAH (cf. FIGURE 3.7).

3. *Protocole entre MN et D :* une clé *Kadhoc-auth* est employée par le protocole EAP-PSK pour effectuer une authentification mutuelle. Dans [62], celle-ci n'est pas nécessaire, il suffit que le MN envoie ses éléments d'autorisation (identifiant, certificat ou jeton) à D afin que celui-ci reconnaisse ses droits. D effectue, à son tour, une comptabilité des consommations de MN en service de contenus (cf. section 3.3.3.2).

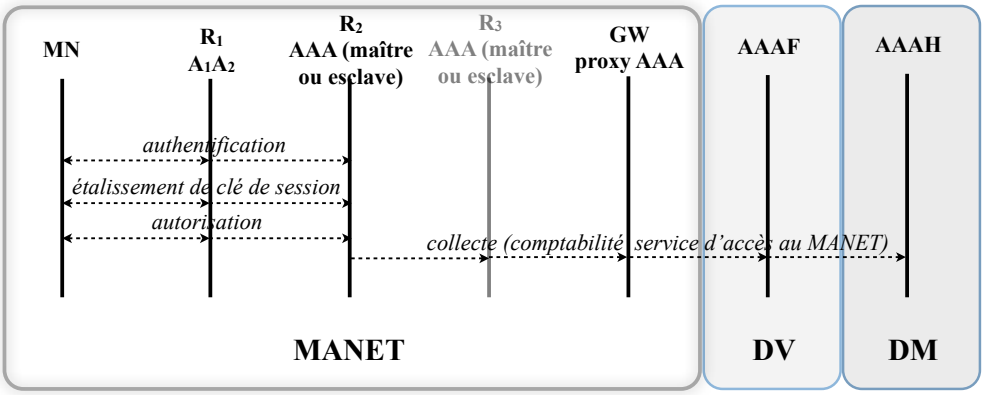


FIGURE 3.8 – MANET partiellement tributaire à architecture AAA hiérarchique centrale : diagramme de séquences

20. Cette méthode est désignée par le nom de méthode de génération dynamique de clés (en anglais *Dynamic key generation*).

**MANET partiellement tributaire à architecture AAA hiérarchique centrale** WATCHMAN [69] a décrit un protocole one-to-one entre MN et un serveur AAA quelconque, maître ou esclave. Les messages de ce protocole sont relayés par $R_1$, une entité $A_1A_2$ (cf. FIGURE 3.8).

MN demande l'accès au réseau. L'authentification est basée sur la cryptographie asymétrique (cf. section 2.4.1 du chapitre 2). $R_1$ demande, alors, à MN sa clé publique que celui-ci lui envoie. $R_1$ transmet cette clé au serveur AAA qui peut être maître ou esclave. Ce dernier tient à jour une base de données des utilisateurs comportant leurs clés publiques et leurs droits à la lumière de laquelle il accepte ou rejette la demande du MN. En cas d'acceptation, $R_1$ procède à l'authentification du MN par un protocole question-réponse (cf. section 2.4.3.1 du chapitre 2). Si celle-ci réussit, $R_1$ fournit des éléments d'accès au MN. Ce dernier point n'a pas été détaillé. D'autre part, $R_1$ envoie l'adresse IP du MN au serveur AAA afin que celui-ci commence les opérations de comptabilité des consommations du MN qui sont donc basées sur le temps de connexion de celui-ci.

**MANET partiellement tributaire à architecture AAA pair-à-pair** Le modèle de confiance dans ce type d'architecture telle que décrit par Lamsal [68] est deux-à-deux c.-à-d. qu'à chaque fois qu'un nœud serveur a accepté la demande d'accès d'un nœud client, une relation de confiance est établie entre eux deux mais pas avec le reste des nœuds ad hoc du réseau.

Toutefois le nœud serveur lui même s'était auparavant authentifié auprès d'un autre nœud serveur (cf. FIGURE 3.9). La confiance est ainsi en chaîne établie de proche en proche. Lamsal n'a pas défini de protocole d'authentification permettant d'établir cette confiance. Néanmoins, ce mode de fonctionnement rappelle celui de Moustafa et al. [64] où des nœuds voisins s'authentifient mutuellement.

#### 3.3.4.2 Protocoles d'authentification one-to-many

Nous avons rencontré ce mode de communication dans le *modèle de serveur de groupe* de Lamsal opérant dans un MANET partiellement tributaire à architecture AAA distribuée.

**MANET partiellement tributaire à architecture AAA distribuée** Ici encore Lamsal n'a pas défini de protocoles. Un serveur AAA consulte un groupe de nœuds avant de permettre l'accès au réseau par le client. Il émet pour cela un message vers chacun des nœuds de ce groupe qui lui répondent ensuite individuellement. Si le serveur décide, à la lumière de ces réponses, que le client est autorisé à accéder au réseau, une relation de confiance entre celui-ci et tous les nœuds du réseau est alors établie. Il n'y a pas besoin d'authentifications ultérieures.

### 3.3.5 Synthèse

Le tableau 3.1 récapitule la classification élaborée dans cette section.

| Types \ Solutions | SAACCESS [62] | DAIDALOS [61] | SESSI [65] | Moustafa et al. [64] | DoCoMo [63] | WATCHMAN [69] | LAMSAL [68] | Sprite [66] |
|---|---|---|---|---|---|---|---|---|
| MANET totalement tributaire | X | X | X | X | X | | | X |
| MANET partiellement tributaire | | | | | | X | X | |
| MANET autonome | | | | | | | | |
| Architecture AAA centrale | | | | | | | | X |
| Architecture AAA hiérarchique centrale | X | X | X | X | X | X | X (2 srvs élus) | |
| Architecture AAA hiérarchique distribuée | | | | | | | X (srvs de groupe) | |
| Architecture AAA pair-à-pair | | | | | | | X (pair-à-pair) | |
| Service par Internet | X | X | X | | | | | |
| Service d'accès à Internet | X | X | X | | | | | |
| Service de contenus | X | | | X | X | | | |
| Service d'accès au MANET | X | X | | | | | | X |
| Protocole one-to-one | X | X | | X | X | X | X (2 srvs élus) (pair-à-pair) | X |
| Protocole one-to-many | | | | | | | X (srvs de groupe) | |

Tableau 3.1 – Taxonomie des solutions AAA dans les MANETs

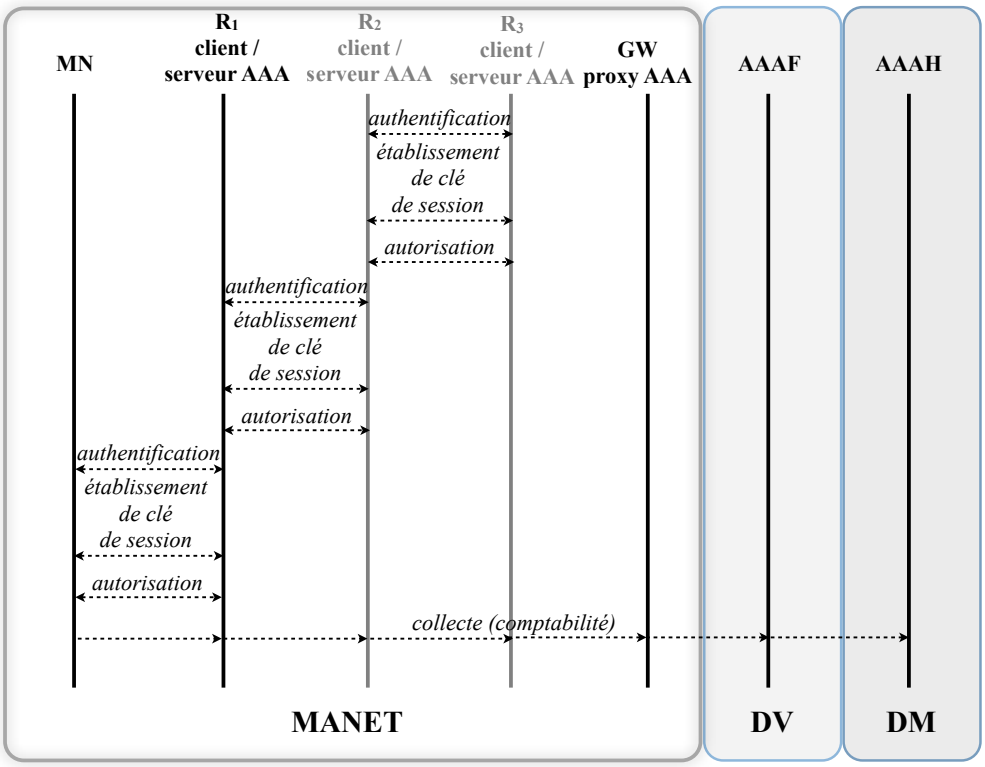


FIGURE 3.9 – MANET partiellement tributaire à architecture AAA pair-à-pair : diagramme de séquences

## 3.4 Considérations annexes

### 3.4.1 Bases de données

Le fonctionnement des trois services AAA nécessitent la disponibilité d'un certain nombre de données de deux sortes : les données persistantes ou invariables et les données variables. Les données invariables sont nécessaires à l'authentification et l'autorisation des clients. Les données variables sont enregistrées au fur et à mesure de l'authentification et de l'autorisation des clients d'une part et de leurs consommations d'autre part. Ces données sont concernés par des problèmes d'initialisation, de mise à jour et de sécurité.

Dans WATCHMAN [69], trois structures de données sont mises en place pour accueillir ces données :

- base de données des clients : elle se trouve dans chaque serveur AAA et contient les informations des clients : identifiants, clés publiques, niveau d'autorisation, etc. Elle est initialisée par l'opérateur dans le MAS (*Master AAA server*),
- table de confiance de cache : chaque nœud ad hoc maintient une telle table. Celle-ci est remplie au fur et à mesure et contient la liste des nœuds déjà authentifiés. Elle est consultée à chaque demande de service par un nœud client,

– informations collectées pour les besoins de la comptabilité : elles sont sous forme de fichiers de journalisation, créés par le MAS et reçues par chaque SAS après son élection. Elles portent sur les durées de connexions des nœuds calculées entre les moments de leur authentification et les dates de leur départ.

Chaque nœud ad hoc dans le système SESSI [65] possède un dépôt sous forme d'une base de données SQL (*Structured Query Language*) contenant les éléments d'accréditation des clients, à savoir leurs clés publiques certifiées par une autorité tierce ou par le nœud lui même. Ce dépôt contient aussi les différentes clés que le client a pu négocier ou dériver après son authentification auprès des services auxquels il a accédé. Il comprend, d'autre part, une liste de contrôle d'accès dans laquelle le nœud définit l'ensemble des nœuds ad hoc ayant le droit d'accéder aux services qu'il propose et leur niveau d'accès. Il contient, d'un autre côté, l'ensemble des services auxquels le nœud a droit.

Dans les modèles de Lamsal [68], pour chaque nœud du réseau, un nœud enregistre dans son cache un triplet portant sur les ressources auxquelles il a droit d'accéder, le type de droit (par exemple lecture ou écriture) et la valeur de la confiance qu'il a en lui. Une valeur entre 0 et 5 signifie une absence de confiance quand elle vaut 0 et une confiance totale quand elle vaut 5. Une politique d'acceptation d'un nœud arrivant peut être fixée. Par exemple, selon le modèle d'architecture AAA, les serveurs et/ou les nœuds collaborant évaluent la demande de ce nœud et la confronte aux triplets le concernant se trouvant dans leurs caches. Ils prennent en compte aussi leurs capacités à prendre en charge le nœud arrivant car ils peuvent décider de préserver leur énergie pour des situation de nécessité.

La solution de Moustafa et al. [64] utilise la table de routage de chaque nœud pour noter les nœuds voisins qui sont authentifiés et ceux qui ne le sont pas. Ainsi avant de faire acheminer son trafic par un voisin, il consulte sa table et ne s'adresse qu'à un voisin authentifié. La solution DoCoMo [63] préconise que l'opérateur d'un nœud distributeur envoie des données contenant son propre certificat et le certificat de chacun des opérateurs avec qui il a conclu un accord. De cette façon, le nœud distributeur peut reconnaître les utilisateurs enregistrés avec ces opérateurs.

### 3.4.2 Architecture matérielle et logicielle

SAACCESS propose une architecture matérielle et logicielle des nœuds ad hoc qui sépare ce qui relève de certains des services AAA et qui doit être contrôlé par l'opérateur, de ce qui relève de services de contenus qui doit être contrôlé par le fournisseur de contenus et, enfin, de ce qui relève de l'utilisation quotidienne du terminal par son propriétaire.

## 3.5 Discussion

L'étude que nous avons menée à la section précédente a permis de mettre en lumière les avancées qui ont été réalisées à ce jour dans la conception de services AAA pour les utilisateurs de MANETs. Elle a montré que les efforts de recherche ont été surtout concentrés sur les MANETs vus comme une extension d'un réseau à infrastructure. En effet, la présence d'un ou plusieurs opérateurs,

parfois désignés simplement par autorité/tiers de confiance ou schématisés par un simple serveur, a été jugée nécessaire pour permettre de créer une relation de confiance entre les nœuds ad hoc et/ou pour collecter les informations pour les besoins de la comptabilité.

Les solutions que nous avons classées dans la catégorie de MANETs totalement tributaires décrivent des architectures AAA précises avec un serveur AAA central hébergé par un opérateur, un ou plusieurs proxy AAA dans le réseau de l'opérateur même ou, aussi, dans le MANET, et des nœuds ad hoc ayant la capacité d'opérer des tâches d'authentification, d'autorisation et/ou de collecte. Ces solutions définissent, par ailleurs, les services commercialisables dans le cadre d'un MANET et la manière de comptabiliser leur consommation et de les facturer. Certaines ont aussi détaillé les protocoles d'authentification à mettre en œuvre entre les différentes parties en proposant d'étendre un protocole AAA tel que Kerberos [64] ou un protocole d'accès comme PANA [61] ou 802.1X [75]. La gestion de clés a été aussi abordée afin de pouvoir mener ces authentifications. Ainsi, l'utilisation de clés symétriques dans Kerberos a été adaptée dans [64], mais nous avons remarqué que l'emploi de couples de clés publiques/privées (à travers l'utilisation de certificats) est beaucoup plus majoritairement employé dans la littérature [62] [65] [63].

Toutefois, cette catégorie des MANETs reste dépendante, pour la majeure partie des opérations, de la présence permanente d'un opérateur. Cela risque de ne pas être toujours possible notamment lorsqu'aucun des nœuds du réseau MANET ne se trouve à proximité d'un point d'accès de cet opérateur. Dans le cas où cette situation dure assez longtemps, l'authentification d'un nœud arrivant devient alors irréalisable et ses droits d'accès aux services inapplicables.

Nous avons vu que ce problème a été traité dans les solutions de la catégorie des MANETs partiellement tributaires qui mettent en œuvre des serveurs AAA dans le MANET même. Néanmoins ces solutions n'ont pas encore atteint un degré d'étude précis semblable à celui des solutions de la catégorie précédente. Lamsal,, par exemple n'a pas défini de protocole et a fait allusion à une possible adaptation du protocole Diameter étant donnée la flexibilité de celui-ci [68]. D'autre part, bien que dans WATCHMAN [69] il ait été précisé un protocole d'authentification entre deux nœuds ad hoc, l'un serveur et l'autre client, le résultat de l'authentification dépend encore de la bonne volonté de ce serveur. À supposé qu'il soit corrompu, il peut empêcher le client d'accéder à certains services auxquels il a droit. De plus, le serveur MAS centralise la tâche d'élection des SAS et les tâches de restitution des données de comptabilité à l'opérateur, ce qui pourrait le placer comme cible de choix pour certains nœuds malveillants. La corruption du MAS pourrait, par exemple, entraîner l'élection de SASs corrompus et complices de celui-ci ou la modification des données de comptabilité provoquant ainsi à une perte financière pour l'opérateur et/ou pour les clients et un gain pour les nœuds malveillants.

Un dernier point, qui nous semble important et qui concerne les deux catégories des MANETs évoquées dans cette section, est celui de l'assurance de l'envoi et de l'intégrité des données de comptabilité : comment pouvons-nous être sûrs qu'un nœud collecteur de ces informations ait la bonne volonté de restituer ces informations, telles qu'elles, au serveur central ? La solution Sprite [66] tente d'apporter des éléments de réponses en ayant recours, dans sa démarche, à des théories économiques qui favorisent le bien-être des utilisateurs.

## 3.6 Conclusion

Dans le premier chapitre, nous avons étudié les concepts et les techniques de sécurité tels qu'on peut les mettre en œuvre dans le contexte d'un réseau à infrastructure. Dans le deuxième chapitre, nous avons présenté les caractéristiques des réseaux ad hoc et mis en évidence le fait que ces solutions de sécurité devaient être réajustées pour tenir compte d'un contexte où la protection doit être prise en charge par les différents nœuds du réseau ainsi que par leurs organisations puisque l'on ne peut plus bénéficier des protections liées à l'infrastructure. Dans ce chapitre, nous avons abordé différentes classifications d'architectures AAA et mis en évidence les solutions connues dans les cas répertoriés.

D'après les points soulevés à la section 3.5, il apparaît qu'il demeure plusieurs aspects non encore étudiés ou détaillés dans la littérature. Parmi ces éléments, nous nous sommes intéressés au problème de centralité du serveur AAA dans le cadre des MANETs partiellement tributaires. Nous chercherons, donc, dans les parties suivantes de cette thèse à résoudre ce problème grâce à la distribution du pouvoir de décision du serveur central sur plusieurs nœuds serveurs. Cela permettra d'éviter, par la même occasion, que le serveur soit un point unique de défaillance.

# Deuxième partie

# Conception de services AAA distribués

# Chapitre 4

# Contribution : architecture, protocoles et opérations AAA distribués

## 4.1 Introduction

Dans ce chapitre, nous examinerons une proposition de solution qui par rapport aux classifications élaborée à la section 3.3 du troisième chapitre se place de la façon suivante :

1. vis à vis du modèle de dépendance, nous sommes dans un MANET partiellement tributaire. La majeure partie des solutions passées en revues dans le troisième chapitre étaient destinées à fonctionner en présence d'un opérateur. Celles qui en étaient partiellement ou complètement indépendantes manquaient de maturité, d'où l'intérêt de nous placer dans ce cadre.
2. vis à vis du modèle d'architecture, nous nous situons dans une architecture hiérarchique distribuée. Ce choix découle du premier point. En effet, l'absence d'un réseau d'opérateur, même momentanée, implique l'absence du serveur AAA central de celui-ci, d'où la nécessité d'un serveur AAA dans le réseau MANET même. Mais l'étude que nous menons à la section 4.2.1 montre qu'il faut en réalité plusieurs serveurs secondés par des entités supplémentaires.
3. vis à vis des services commercialisables, notre solution AAA protège l'accès aux services de routage et de contenus en supposant que tout nœud authentifié y est autorisé.
4. enfin, vis à vis du mode de communication, le deuxième point nous conduit à privilégier majoritairement un mode de communication one-to-many.

La première section a pour objectif d'établir le cahier des charges des services AAA que nous souhaitons offrir dans le cadre d'un MANET partiellement tributaire. Les spécifications de l'architecture AAA, des protocoles qu'elle mettra en œuvre et des aspects de sécurité qu'elle doit respecter y sont en particulier précisées. Afin de répondre à ces spécifications, les sections suivantes présentent nos propositions d'architecture AAA et de protocoles d'authentification et d'autorisation. Un bref aperçu des opérations de collectes en vue d'une comptabilité est

aussi donné à la dernière section avant la conclusion. Enfin, en ce qui concerne les aspects de sécurité, ils sont traités tout au long de ces sections.

Pour suivre les tendances actuelles visant à contourner le problème de pénurie d'adresses IPv4, nous supposons tout au long de ce chapitre que les réseaux considérés utilisent le protocole IPv6 [1] comme protocole de couche réseau.

## 4.2 Cahier des charges

Nous avons établi un cahier des charges tenant compte des caractéristiques des MANETs et des exigences raisonnables de sécurité dans ce type de réseau.

### 4.2.1 Architecture

Nous souhaitons que le modèle d'architecture soit distribué, car bien qu'assigner à un nœud unique le rôle de serveur AAA soit la solution la plus simple et la plus intuitive, le fonctionnement du système reposera alors entièrement sur ce seul serveur. Cette approche n'est donc pas tolérante aux fautes et l'architecture du système y est très vulnérable : il suffit que le serveur tombe en panne ou soit l'objet d'une attaque par un nœud malveillant pour que tout le système AAA s'écroule. De plus, la disponibilité du service repose sur celle de ce seul serveur qui peut quitter le réseau, passer en mode veille, se trouver en train de traiter un nombre de requêtes important qui dépasse ses capacités physiques, etc.

D'autre part, distribuer les services AAA entre plusieurs serveurs ne signifie pas la réplication du serveur unique du modèle ci-dessus en plusieurs serveurs identiques. En effet, si $n$ est le nombre de ces serveurs, la réplication permettrait, certes, de résoudre le problème d'indisponibilité puisque n'importe lequel des serveurs serait capable de fournir les services AAA. Elle permettrait aussi d'améliorer la tolérance du système aux pannes parce qu'il serait devenu capable de supporter jusqu'à $n - 1$ serveurs en panne. Mais, cette amélioration de la tolérance aux pannes s'accompagne d'un accroissement de la vulnérabilité du système car il suffit pour un attaquant de corrompre un seul serveur, pour entrer en possession de toutes les données du système, en particulier les données secrètes que sont les clés cryptographiques. Il apparaît donc que la réplication n'est pas une solution utilisable dans un MANET.

Afin que le système soit robuste, il vaut mieux que les données et les opérations puissent être majoritairement réparties sur plusieurs serveurs. Cette distribution peut être réalisée de diverses manières choisies selon les nœuds concernés par la fourniture du service AAA, le type de données secrètes, les mécanismes d'authentification, d'autorisation et de collecte pour les besoins de comptabilité et divers autres paramètres dont l'importance relative apparaîtra par la suite.

### 4.2.2 Besoins et choix protocolaires

Nous souhaitons que les protocoles d'authentification, d'autorisation et de collecte, qui font évidemment partie de l'ensemble des opérations des serveurs,

1. Un des apports importants de la sixième version du protocole IP, mis à part l'augmentation de la plage d'adresses, est les mécanismes de configuration automatique des adresses IPv6.

soient distribués majoritairement entre les serveurs et, si besoin, d'autres nœuds du réseau.

En ce qui concerne le protocole d'authentification, il doit reposer sur le protocole EAP par la définition d'une méthode d'authentification adéquate. L'intégration de cette méthode avec des protocoles AAA existants, comme RADIUS ou Diameter, sera de cette manière possible ainsi que nous l'avons évoqué à la fin de la section 3.2.3.3 du chapitre 3.

D'autre part, cette méthode d'authentification ne doit pas permettre les attaques de type déni de services. Elle doit aussi assurer une authentification mutuelle entre le système fournissant les services AAA et les nœuds arrivants souhaitant accéder aux services souscrits. En effet, alors que, dans le contexte d'un réseau à infrastructure, l'authentification mutuelle n'est pas toujours indispensable entre un utilisateur et le serveur AAA de son opérateur, dans un contexte à risque comme celui des réseaux MANETs (cf. section 1.4 du chapitre 1), celle-ci devient impérative.

Enfin, les protocoles conçus doivent pouvoir fonctionner dans un réseau MANET utilisant la norme IPv6. Celle-ci apporte non seulement des solutions au problème de pénurie d'adresses IPv4 mais aussi au problème de configuration par les nœuds eux-même de leurs adresses IPv6 (cf. section C.1 de l'annexe C). Il est clair que cette capacité, dotée en plus d'un mécanisme permettant d'éviter le vol d'adresses, ne peut qu'être appréciée dans le cadre des MANETs.

### 4.2.3 Outils cryptographiques

Les algorithmes cryptographiques nécessaires à la sécurité des opérations d'authentification, d'autorisation et de comptabilité doivent être conçus pour permettre un fonctionnement distribué entre les différents serveurs.

Ils doivent par ailleurs être relativement robustes et peu consommateurs de ressources compte tenu des capacités, souvent limitées des nœuds mobiles, qu'il s'agisse de puissance de calculs, d'espaces mémoires ou d'énergie des batteries (cf. section 1.4.7 du chapitre 1). Les messages obtenus après utilisation de ces outils ont aussi intérêt à être courts afin de limiter l'énergie en émission et en réception.

En plus de garantir l'authentification des différentes parties, la confidentialité de leurs échanges en cas de besoins et l'intégrité de leurs données, les algorithmes et les méthodes cryptographiques choisies doivent parer l'attaque de l'adversaire mobile [76]. Cette dernière consiste à approcher suffisamment de plusieurs nœuds successifs pour obtenir de chacun d'eux des éléments dont l'importance paraît individuellement faible mais dont la conjonction permet une attaque précise contre les outils cryptographiques : la mobilité permet ainsi ce qui est beaucoup plus difficile, voire inaccessible avec les réseaux filaires.

## 4.3 Architecture proposée

### 4.3.1 Description et vocabulaire

Deux options se sont présentées à nous pour satisfaire le cahier des charge : distribuer un serveur AAA sur quelques uns des nœuds ad hoc du réseau ou bien le distribuer sur tous les nœuds. Dans la mesure où les protocoles doivent aussi

être distribués amenant plusieurs serveurs, à émettre en même temps, il est clair que le deuxième choix engendre plus de messages que le premier et consomme, donc, une part plus importante de la bande passante du réseau. Nous avons, par conséquent, opté pour la première option : seuls quelques nœuds ad hoc sont désigner pour jouer le rôle de serveurs AAA.

Cependant, en raison de l'organisation sans infrastructure et mobile des MANETs, l'entrée des clients dans le réseau n'est pas toujours réalisable à partir d'un point spécifique, par exemple un ou plusieurs nœuds bien identifiés. Il est donc nécessaire d'impliquer tous les nœuds dans les opérations de contrôle d'accès. Donc, tout nœud ad hoc, serveurs AAA compris, doit jouer le rôle d'une entité $A_2$ (cf. les notations de la section 3.3.2 du chapitre 3) capable de vérifier le niveau d'autorisation d'un nœud entrant. Il est possible de satisfaire cette contrainte si chaque nœud est conçu comme un client AAA supportant un *Enforcement Point* (cf. section 3.2.1 du chapitre 3).

Par ailleurs, un nœud qui souhaite accéder au réseau joue classiquement le rôle de *client* (cf. section 3.2.1 du chapitre 3). Ainsi, dans notre architecture, il jouera, à la fois, le rôle de *client* et de *client AAA*, deux rôles *à priori* différents. Dans le modèle d'architecture AAA décrit à la section 3.2.1 du chapitre 3, le client AAA se trouve entre le *client* et le serveur AAA. Il transmet les demandes d'accès du premier vers le second. Pour pouvoir confondre les rôles de *client* et celui de client AAA dans un nœud donné, il suffit, donc, que le client AAA, au sein de ce nœud, génère ses propres demandes d'authentification, opération déjà supportée dans le modèle d'architecture AAA initial. Remarquons que de cette manière il n'est plus nécessaire pour un nœud *client* de recourir à un client AAA hébergé sur un autre nœud intermédiaire pour envoyer ces demandes d'accès aux serveurs. Ce fonctionnement est appréciable car un nœud intermédiaire serait une cible toute désignée pour un attaquant. D'ailleurs, nous avons déjà cherché à éviter ce type de problème en ce qui concerne l'utilisation du serveur AAA lui-même en le distribuant sur plusieurs nœuds.

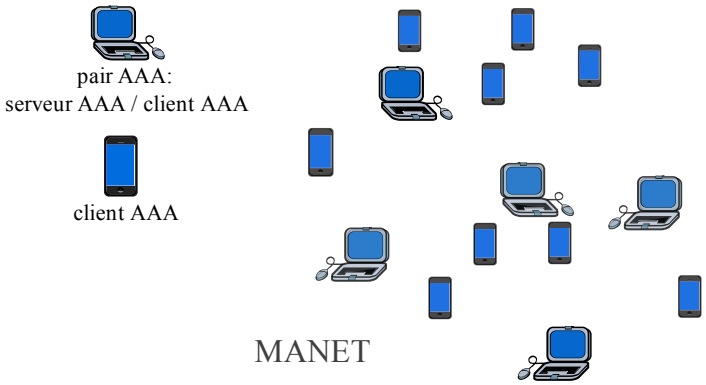


FIGURE 4.1 – *Dist-AAA* : Architecture AAA destinée à fonctionner dans un MANET

La figure 4.1 montre les différentes entités qui entrent en jeu dans *Dist-AAA*, l'architecture AAA que nous avons proposée. Il est à remarquer qu'un

serveur AAA joue aussi le rôle de client AAA. Nous l'appellerons donc *pair AAA* ou tout simplement *pair*. Pour les mêmes raisons de simplification de langage, nous désignerons un client AAA par le nom *client*. Nous sommes conscients de l'ambiguïté de langage que cela entraîne, c'est pourquoi le tableau 4.1 vient rappeler et résumer ces notations.

| notation | description |
|---|---|
| *Dist-AAA* | Nom de l'architecture AAA hiérarchiquement distribuée que nous avons définie |
| *client* (AAA) | Tout nœud ad hoc du réseau. Il est *client AAA* au sens du service AAA fonctionnant en mode client/serveur. Il est capable d'initier ses propres demandes d'authentification. Par ses fonctions d'EP (*Enforcement Point*), il renforce aussi les politiques de contrôle d'accès (cf. section 3.2.1 du chapitre 3) en appliquant des règles de filtrage au trafic à relayer |
| *pair* (AAA) | un nœud ad hoc supportant à la fois les fonctions d'un *client AAA* et d'un *serveur AAA*. Il est client AAA quand il s'agit de s'authentifier lui-même pour accéder au réseau et lorsqu'il s'agit de renforcer les politiques de contrôle d'accès. Il est *serveur AAA* lorsqu'il s'agit d'authentifier un *client* en collaborant avec d'autres serveurs |

Tableau 4.1 – Résumé des notations dans *Dist-AAA*

Les clients viennent seconder la tâche des pairs en renforçant la politique de contrôle d'accès grâce à leur fonction d'EP. Il apparaît donc que cette architecture est du type *hiérarchique distribué*, au sens de la section 3.3.2.4 du chapitre 3 : les pairs sont en haut de l'architecture et les clients sont des entités subordonnées.

## 4.3.2 Initialisation

Bien que cette thèse ne traite pas du problème de l'initialisation de l'architecture AAA, cette question n'en est pas moins pertinente. On s'en tiendra pour les réponses possibles aux éléments et solutions recensés au chapitre 3.

### 4.3.2.1 Désignation des pairs AAA

Nous avons vu à la section 3.3.2.2 du chapitre 3 qu'il existe deux procédés de désignation des serveurs :

1. l'élection : des élections sont organisées par tous ou certains nœuds ad hoc du réseau pour sélectionner ceux parmi lesquels le rôle de pairs pourra être joué. Les élections se basent sur un ensemble de critères identiques pour tous les nœuds. Ces critères peuvent porter sur les capacités physiques (puissance de calculs du processeur, espace de mémoire, énergie) des terminaux mobiles, sur leur degré de mobilité, sur leur localisation, sur le niveau de confiance qui leur a été accordé par les autres nœuds, etc. Parmi les travaux plus spécifiques traitant de ce problème, on peut citer [77] et [78].
2. la désignation par un tiers de confiance : un tel tiers peut être constitué par un ou plusieurs opérateurs agissant conjointement, un fournisseur de contenus, etc. Lui aussi se base sur un ensemble de critères dans sa sélection des serveurs. Certains critères peuvent être communs avec ceux

utilisés pendant les élections tels que les capacités physique ou le degré de confiance qui est, bien entendu, établi par le tiers de confiance et non les nœuds du réseau.

#### 4.3.2.2 Initialisation des bases de données

**Éléments d'accréditation** Dans les premiers chapitres de cette thèse, nous avons établi que les éléments d'accréditation étaient à la base de la constitution de toute relation de confiance et de tout contrôle d'accès (cf. sections 2.3.3, 2.4.3.1 et 2.5 du chapitre 2 et sections 3.2.3 et 3.3.1.1 du chapitre 3). Dans notre proposition, comme nous sommes dans un cas de MANETs partiellement tributaire, ces éléments sont des certificats publics émis par le tiers de confiance. Ainsi, tout nœud ad hoc est un client qui possède une clé privée associée à une clé publique. Cette dernière est liée à l'identité et à un certain nombre d'informations sur le client, toutes inscrites dans un certificat signé par une autorité placée sous l'administration du tiers de confiance [2]. Dans notre travail, la forme des certificats utilisés suit la norme X.509 [79]. Nous les désignerons par certificats X.509 [3].

**Bases de données** Comme dans la section 3.4.1 du chapitre 3, nous optons pour deux sortes de bases de données installées dans le terminal de chaque nœud :

1. une *base des utilisateurs* : une base de données contenant l'identifiant, les éléments d'accréditation et le niveau d'autorisation de chaque nœud du réseau. Les identifiants et les niveaux d'autorisation sont initialisés par le tiers de confiance alors que les éléments d'accréditation, c.-à-d. les certificats, sont découverts et ajoutés à la volée au fur et à mesure de l'arrivée des nœuds dans le réseau.
2. une *base des utilisateurs autorisés* : une base de données contenant les données d'autorisation, notamment des jetons d'accès dans notre proposition (cf. section 4.5.1), et celles de comptabilité où les éléments d'autorisation et les ressources consommées par un client sont renseignées dans une entrée contenant au départ son identifiant. Les données de comptabilité sont transmises régulièrement aux serveurs AAA.

Il serait possible de penser que de telles bases puissent occuper un volume conséquent de l'espace disque de chaque nœud. Pourtant dans un réseau d'une centaine de nœuds, la taille de chacune des deux bases serait de l'ordre de 100 Ko si l'on utilisait tout simplement un tableau. C'est par exemple la taille d'un tel tableau géré sur le mode d'une feuille de calcul dans un logiciel courant comme Excel de Microsoft ou Numbers d'Apple. Elle serait plus petite encore

2. En effet le tiers de confiance inclut tout un ensemble de services et d'outils qui lui permettent de jouer ce rôle à divers moments du processus AAA (cf. section 3.2.3 du chapitre 3).

3. Un certificat X.509 comporte plusieurs champs, dans l'ordre : le numéro de version de la norme, pour nous la version 2 ; le numéro de série ; l'algorithme de signature de certificat, pour nous RSA ; le nom de domaine de l'autorité de certification, pour nous c'est le même que celui du tiers de confiance ; date limite de validité ; le domaine du détenteur du certificat, il peut être différent de celui du tiers de confiance notamment s'il s'agit de plusieurs opérateurs ; information sur la clé publique, notamment son algorithme, dans notre cas c'est RSA ; d'autres champs optionnels et la signature de l'autorité de certification.

en utilisant les outils intégrés dans le système iOS d'un iPhone. Si la taille d'un certificat peut être estimé à 4 Ko, soit 400 Ko pour 100 nœuds, un volume de données d'environ 600 Ko par nœud sera alors à stocker. Sachant que la capacité totale du disque d'un iPhone de 3ème génération est de 7 Go, cela représente 0,0008 %, autant dire que c'est négligeable.

### 4.3.3 Évolution

Au cours de la vie du réseau, et après la phase d'initialisation, d'autres nœuds peuvent rejoindre le réseau ou le quitter. Le nombre de pairs AAA peut évoluer lui aussi en fonction du nombre total de nœuds dans le réseau. Un nombre insuffisant de pairs peut entraîner l'indisponibilité du service AAA. Au contraire un trop grand nombre entraînerait un gaspillage de la bande passante. Nous donnerons des exemples de tels dis-fonctionnements tout au long de ce chapitre et du chapitre suivant (cf. en particulier la section 7.2 du chapitre 7). Cela étant noté, nous n'effectuerons pas d'études permettant de trouver le nombre adéquat de pairs AAA dans un réseau donné.

Fred,
*Le Naufragé du « A » :*
découverte du premier A

## 4.4 Protocole d'authentification proposé

Avant de décrire un protocole d'authentification de notre conception répondant aux exigences définies par les sections 4.2.2 et 4.2.3, nous présentons les briques élémentaires permettant de l'élaborer. L'authentification mutuelle par le protocole ISO 9798-3 et la distribution par l'emploi de la cryptographie à seuil seront examinées tout particulièrement.

Ensuite, la description du protocole est effectuée en trois parties correspondant aux phases de conception par lesquelles nous sommes passées :

1. intégration des briques élémentaires permettant d'aboutir à un protocole en six étapes,
2. optimisation du nombre d'étapes à quatre étapes,
3. prise en compte de l'échec d'une opération d'authentification en multipliant le nombre de tentatives d'authentification suivant un algorithme appelé *algorithme à multiples tentatives d'authentification.*

### 4.4.1 Authentification mutuelle avec ISO 9798-3

Le standard ISO 9798-3 [80] fournit une méthode d'authentification qui assure l'authentification de deux entités par l'envoi de leurs certificats respectifs et l'échange de leurs signatures suivant le modèle question-réponse (cf. section 2.4.3.1 du chapitre 2). Le certificat reçu peut, par la suite, être consigné dans la base des utilisateurs de l'entité destinatrice (cf. section 4.3.2.2). L'emploi des signatures assure la non-répudiation (cf. section 2.4.3.4 du chapitre 2).

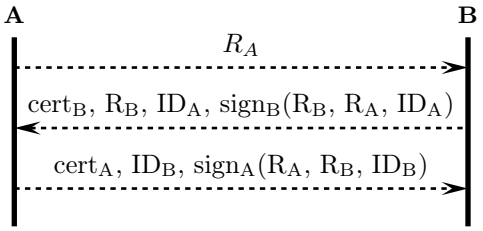


FIGURE 4.2 – Authentification mutuelle du standard ISO 9798-3

Le protocole d'authentification du standard ISO 9798-3 est présenté par la figure 4.2. Il s'agit d'un protocole à trois messages. D'abord la partie initiatrice des échanges, A, envoie un nombre aléatoire $R_A$ à la partie destinatrice, B. Celle-ci choisit un autre nombre aléatoire, noté $R_B$, signe ($R_B$, $R_A$, $ID_A$) où $ID_A$ est l'identifiant de A, et envoie la signature obtenue, ainsi que son certificat $cert_B$ et les informations $R_B$ et $ID_A$ à A. Cette signature a pour but de prouver à A l'authenticité de B. Les nombres aléatoires figurant dans cette signature sont choisis conjointement par A et B afin d'éviter que A ne fasse chiffrer par B une information de son choix, menant ainsi une attaque par texte clair [4] pour découvrir la clé privée de B. Le certificat de B est ensuite utilisé par A, après vérification de son authenticité [5], pour récupérer la clé publique de B et déchiffrer la signature reçue. Si le résultat est égal au condensat de ($R_B$, $R_A$, $ID_A$), l'authentification de B par A est alors réussie. Le troisième message contient une signature générée par A à destination de B afin que celle-ci puisse, de la même manière, authentifier A.

### 4.4.2 Distribution

La méthode d'authentification étant dépendante, en général, des opérations cryptographiques, il suffit, donc, que les algorithmes cryptographiques soient, eux-mêmes, exécutés de façon distribuée. A cette fin, nous avons eu recours à la cryptographie à seuil basée sur RSA. Celle-ci permet de distribuer la tâche d'authentification entre plusieurs pairs sans que chacun utilise une même clé privée. A la fin des opérations, une signature *globale* et unique du système est obtenue. Comme requis, les rôles des pairs pendant la phase d'authentification se trouvent ainsi complémentaires. Un aperçu du fonctionnement de la cryptographie à seuil est donné ci-dessous. L'annexe B donne des éléments de compréhension supplémentaires plus précis.

#### 4.4.2.1 Cryptographie à seuil

Certaines organisations sont gérées par plusieurs mandataires. Pour certaines décisions, particulièrement importantes (utilisation de fortes sommes, engagements financiers ou techniques lourds, déclenchement d'un feu nucléaire,...), il est souhaité qu'elles soient *cosignées* par plusieurs des mandataires, usuellement deux ou trois. C'est le problème de la *co-émission*. On pourrait imaginer que plusieurs exemplaires de la décision chacune signée individuellement par un mandataire puissent être acheminées, mais la lourdeur d'une procédure qui multiplie les clés de chiffrement fait que l'on ne souhaite pas procéder ainsi.

De même, on peut souhaiter qu'un message émis par un signataire unique ne puisse être déchiffré que si un certain nombre de destinataires sont présents. C'est le problème de la *co-réception*. L'un et l'autre problèmes peuvent

4. C'est un type d'attaque où l'adversaire a la possibilité de choisir un texte spécifique à faire chiffrer par une partie afin d'obtenir une information supplémentaire sur sa clé secrète privée.

5. A doit connaître la clé publique de l'autorité de certification de B pour établir cette authenticité. En pratique, A possède un ensemble de certificats des autorités les plus connues et est en mesure, grâce à la relation hiérarchique qui les organise, de connaître et authentifier leurs clés publiques. Grâce à la clé publique de l'autorité de certification de B, A vérifie, en la déchiffrant, que la signature apposée au certificat de B a bien été générée par cette autorité.

se résoudre par des méthodes de cryptographie à seuil (en anglais *threshold cryptography*).

Dans la cryptographie à seuil, un secret partagé entre plusieurs entités permet à un sous ensemble d'entre elles, de cardinal égal à un nombre seuil, de cosigner un message. C'est une méthode très appréciée dans les systèmes distribués car elle permet d'éviter le problème de *point individuel de défaillance* (en anglais *single point of failure*[6]).

On appelle « schéma cryptographique seuil $(n, th)$ » tel que $1 \leq th \leq n$ utilisé par un système à $n$ entités, un schéma vérifiant les conditions suivantes :

1. un secret du système est partagé en $n$ *parts de secret*. Chaque part est détenue par une entité du système[7],
2. la connaissance de $th$ parts ou plus permet de retrouver le secret du système,
3. la connaissance de $th - 1$ ou moins de parts ne permet en aucune façon de trouver le secret parce que toutes les valeurs possibles sont également probables,
4. un sous ensemble quelconque d'entités, dont le cardinal est supérieur ou égal à $th$, est capable de cosigner un message donné $M$.

Les points 2 et 3 impliquent que le système, muni de ce schéma, peut supporter jusqu'à $th - 1$ entités corrompues. Il est, par ailleurs, clair que pour que le système continue à fonctionner lorsqu'il y a $th - 1$ entités corrompues, il faut : $th \leqslant \lfloor \frac{n+1}{2} \rfloor$. Concernant le point 4, il est impératif qu'au moment de la *co-signature*, le secret ne soit calculé par aucune entité. Sinon une entité deviendrait capable de générer individuellement des signatures valides.

Shamir [49] a conçu une méthode de partage de secret répondant aux descriptions des points 1, 2 et 3. Elle a été, ensuite, exploitée dans des travaux portants sur le problème de co-émission. Shoup [81] a, en particulier, mis en place un système de co-signature basé sur RSA et répondant à la description du point 4. Ces deux méthodes sont présentées ci-dessous.

**Partage de secret** Dans la bande dessinée *Le secret de la licorne*, Hergé montre une méthode basique de partage de secret et de sa reconstitution par la suite (cf. FIGURE 4.3). Le secret, lieu où se cache le trésor du pirate Rackam le Rouge, s'écrit sous forme d'une chaîne de caractères composée d'un nombre exprimant la latitude, puis la lettre $\mathcal{N}$, puis un nombre exprimant la longitude, enfin la lettre $\mathcal{W}$. Trois suites de caractères prises au hasard dans cette chaîne sont ensuite inscrites respectivement sur trois parchemins en préservant l'emplacement de chaque caractère dans le secret. Ainsi, une fois ces parchemins sont trouvés, ils sont superposés pour retrouver le secret.

Dans le partage de secret par Shamir [49], le calcul des parts de secret et la reconstitution se font grâce à l'emploi de polynômes sur l'anneau $\mathbb{Z}_\nu$ des entiers modulo $\nu$ (cf. section B.1 de l'annexe B).

On suppose qu'Alice désire partager entre les $n$ membres $B_1, B_2, \ldots, B_{n-1}$ de la famille Bernard un secret de la taille d'un tronçon soit un entier $s < \nu$.

6. Le point individuel de défaillance est un point particulier du système qui, s'il s'arrêtait, impliquerait l'arrêt du système en entier.

7. Le cas où certaines entités détiennent plus d'une part est aussi envisageable. Ainsi, selon le nombre de parts possédées, un pouvoir plus ou moins grand est conféré à une entité.

FIGURE 4.3 – Hergé, *Le Secret de La Licorne*, (Les Aventures de Tintin). Superposition des trois parchemins et résolution de l'énigme : les coordonnées de l'île où se cache le trésor du pirate Rackham le Rouge sont ainsi découverts

Alice désire par ailleurs que $th$ quelconques des $B_i$ puissent reconstituer $s$ mais que cela ne soit pas possible à $th-1$ d'entre eux. Pour faire cela, Alice va tout d'abord construire un polynôme de degré $th-1$ dont la valeur en zéro est $s$. En pratique, elle peut choisir au hasard des entiers $a_1, \ldots, a_{th-1}$ tous inférieurs à $\nu$ et définir le polynôme $P$ par $P(X) = s + a_1X + a_2X^2 + \cdots + a_{th-1}X^{th-1}$.

Alice va ensuite acheminer (par un canal sûr) à chacun des $B_i$ sa part de secret constitué de son numéro $i$ et de la valeur $P(i)$, soit $(i, b_i)$.

Maintenant lorsqu'une partie formée de $th$ membres $B_{i_1}, B_{i_2}, \ldots, B_{i_{th}}$ de la famille Bernard a besoin de connaître le secret, elle se réunit et reconstruit le polynôme $P$ en utilisant le polynôme d'interpolation de Lagrange (cf. section B.1.2 de l'annexe B) puis calcule $P(0)$ [8].

Mais faire ce calcul implique que les membres du groupe s'accordent sur le choix de l'un d'entre eux qui les effectuera et pour cela admettent de remettre leurs éléments de secret à celui-ci. Bien sûr on peut facilement imaginer des procédures d'échanges permettant de garantir à chacun des membres d'être en possession des mêmes informations. Néanmoins, l'élu se trouve en situation de connaissance du secret qu'il pourra réutiliser plus tard, sans nécessairement consulter à nouveau les autres membres du groupe. Par exemple si $s$ est une clé privée, il pourra désormais signer tout nouveau message sans référence aux autres membres du groupe.

#### 4.4.2.2 Cryptographie à seuil basée sur RSA

**Partage d'une clé privée RSA** Pour mettre en place un schéma cryptographique seuil $(n, th)$ dans *Dist-AAA* où $n$ est le nombre des pairs AAA, un tiers de confiance, par exemple un opérateur, intervient pendant la phase d'inscription (cf. section 3.2.3.1 du chapitre 3). Il fixe, comme dans la section B.5 de l'annexe B, $p$ et $q$ et calcule $\nu = pq$. Partant de $e$, il détermine l'*exposant privé*, $u$, inverse de l'*exposant public*, $e$, modulo $\varphi(\nu)$. Il donne alors à chacun

8. En fait, les calculs se simplifient un peu car on recherche les $L_i(0)$ qui sont des nombres et non les $L_i$ qui sont des polynômes.

des nœuds pairs AAA une *part de clé privée*. Pour cela, il traite la quantité $u$ comme le secret à partager dans le paragraphe ci-dessus et envoie à chaque pair le triplet $(\nu, i, b_i)$. On sait alors que $u = P(0)$.

**Calculs de parts d'une signature RSA** Soit $M$ l'information à signer par les pairs. Désormais, pour signer un tronçon de $M$ (cf. section B.3 de l'annexe B) dont le haché (cf. section 2.4.2 du chapitre 2) est $m$, chaque pair calcule $c_i = m^{b_i}$. On appelle cette quantité *part de signature* ou *signature partielle* du pair numéro $i$. Elle ne permet pas de retrouver $b_i$ en raison de la complexité du problème du logarithme discret. Ainsi, les $b_i$ restent la propriété exclusive des pairs associés.

**Combinaison des parts de signature** Les parts de signature sont envoyés à une entité, par exemple un client [9], qui les combinera afin de trouver la signature globale du groupe des pairs AAA. Pour cela, il suffit de $th$ parts de signature que nous supposerons, sans perte de généralité, être $c_1$, $c_2$,$\cdots$, $c_{th}$. En utilisant la section B.1.2, l'entité peut déterminer les $L_i(0)$ et calculer [10] :

$$\prod_{i=1}^{th} (c_i)^{L_i(0)} = m^{\sum_{i=1}^{th} b_i L_i(0)} = m^{P(0)} = m^u \tag{4.1}$$

On sait que $m^u$ est une signature qu'aurait pu générer une entité détenant la clé privée ($\nu$, $u$) associée à la clé publique ($\nu$, $e$). C'est, en fait, une signature globale des pairs AAA sur le tronçon $m$. Pour vérifier sa validité, il suffit de vérifier l'égalité $(m^u)^e = m \ mod \ \nu$.

### 4.4.3 Proposition

Grâce aux caractéristiques du protocole présenté à la section 4.4.1 et à la technique de la cryptographie à seuil vue à la section 4.4.2, nous avons défini une nouvelle méthode d'authentification EAP, que nous avons appelée EAP-ADHOC et qui a fait l'objet d'une publication à SETOP, un atelier ayant eu lieu au sein de la conférence SAR-SSI en 2008 [82]. Cette méthode réunit les aspects de chacune de ces techniques et répond ainsi au cahier des charges (cf. section 4.2). Elle consiste, essentiellement, en un protocole semblable au protocole d'authentification mutuelle ISO 9798-3 mais qui transporte une signature partagée des pairs AAA au lieu d'une signature RSA simple. Elle inclus, entre autres, des solutions aux problèmes induits par la mise de ce protocole en mode de communication one-to-many. Enfin, elle prend en considération certains aspects de sécurité et prépare la phase d'autorisation par la livraison d'un jeton d'accès.

La figure 4.4 illustre les échanges qui ont lieu entre un nœud arrivant (en anglais *Joining Node* ou JN) et les pairs AAA. L'ensemble des notations qui y sont employées est résumé dans le tableau 4.2. On suppose que les pairs sont au nombre de $n$. On les note : $\mathrm{srv_1}$, $\mathrm{srv_2}$,$\cdots$, $\mathrm{srv_n}$ et $\mathcal{P} = \{\mathrm{srv_1}, \mathrm{srv_2}, \cdots, \mathrm{srv_n}\}$. Soit $th$ le seuil cryptographique. La réponse d'un nombre de pairs supérieur ou égal

9. Nous verrons que cela sera le cas lorsque la signature globale du groupe des pairs AAA servira à les authentifier.

10. Dans les $L_i(X)$, il suffit de remplacer les $x_i$ par $i$ pour $i \in \{1, \cdots, n\}$

| notation | description |
|---|---|
| EAP-ADHOC | méthode d'authentification que nous avons définie dans le cadre de *Dist-AAA* |
| $n$ | nombre de pairs AAA |
| $th$ | seuil cryptographique |
| JN | *Joining Node*, un nœud arrivant, *client* dans *Dist-AAA* |
| NN | *Neighbor Node*, un nœud voisin de JN |
| $\mathrm{srv_j}$ | pair AAA d'indice $j$ |
| $\mathcal{P}$ | ensemble des pairs $\mathrm{srv_1}$, $\mathrm{srv_2}$,$\cdots$, $\mathrm{srv_n}$ |
| $\mathcal{P}_\kappa$ | ensemble des pairs $\mathrm{srv_{i_1}}$, $\mathrm{srv_{i_2}}$,$\cdots$, $\mathrm{srv_{i_{th}}}$ participant à toutes les étapes du protocole |
| $\mathrm{ID_{JN}}$ | identité du nœud JN |
| $\mathrm{ID_j}$ | identité du pair $\mathrm{srv_j}$ |
| $\mathrm{ID_{AAA}}$ | identité de l'ensemble des pairs AAA |
| $\mathrm{R_j}$ | nombre aléatoire choisi par le pair $\mathrm{srv_j}$ |
| $\mathrm{R_{JN}}$ | nombre aléatoire choisi par JN |
| $\mathrm{cert_{JN}}$ | certificat du JN |
| $\mathrm{cert_{AAA}}$ | certificat de l'ensemble des pairs AAA |
| $\mathrm{sign_{JN}}(p_1, \cdots, p_m)$ | signature du JN appliquée aux paramètres $p_1, \cdots, p_m$ |
| $\mathrm{sign_j}(p_1, \cdots, p_m)$ | signature partielle du pair $\mathrm{srv_j}$ appliquée aux paramètres $p_1, \cdots, p_m$ |
| $\mathrm{sign_{AAA}}(p_1, \cdots, p_m)$ | signature du groupe des pairs AAA appliquée aux paramètres $p_1, \cdots, p_m$ et obtenue par co-signature d'au moins $th$ pairs |
| $\mathrm{T_{JN}}$ | date limite de validité du jeton du JN au delà de laquelle celui-ci doit se ré-authentifier |

TABLEAU 4.2 – Résumé des notations

à $th$ à chaque sollicitation provenant d'un JN est nécessaire. On suppose, alors, qu'un ensemble $\mathcal{P}_\kappa$ [11] de $th$ pairs participent à tous les échanges du protocole. Sur la figure, $th = 3$ et $n$ est un nombre supérieur à $th$. Les échanges sont illustrés entre un JN et $th$ pairs car cela est suffisant pour comprendre le principe du protocole. Sans perte de généralité, de tels pairs peuvent être $\mathrm{srv_1}$, $\mathrm{srv_2}$ et $\mathrm{srv_3}$, donc $\mathcal{P}_\kappa = \{\mathrm{srv_1}, \mathrm{srv_2}, \mathrm{srv_3}\}$.

La succession des échanges suit alors les étapes suivantes :

**Étape 1** JN initie les échanges en envoyant une requête à chaque pair AAA d'un ensemble $\mathcal{P}_1$ (sur la figure $\mathrm{srv_1}$, $\mathrm{srv_2}$ et $\mathrm{srv_3}$ et d'autres pairs en plus) tel que $\mathcal{P}_\kappa \subseteq \mathcal{P}_1 \subseteq \mathcal{P}$. Les pairs peuvent, par exemple, être choisis aléatoirement dans $\mathcal{P}$. Si le JN connaît le nombre de sauts qui le séparent des pairs, il peut aussi choisir ceux qui lui sont les plus proches.

Chaque requête est un message EAP de type identité (sur la figure *Type= Identity*) contenant l'identité du JN, $\mathrm{ID_{JN}}$.

L'étape 1 n'existe pas dans les spécifications de la norme ISO 9798-3 (cf. section 4.4.1). Elle sert, d'une part, au JN comme étape de lancement de la demande d'authentification. D'autre part, en évitant que le JN soit le

11. $\kappa$, du grec $\kappa\alpha\rho\delta\iota\acute{\alpha}$, cœur.

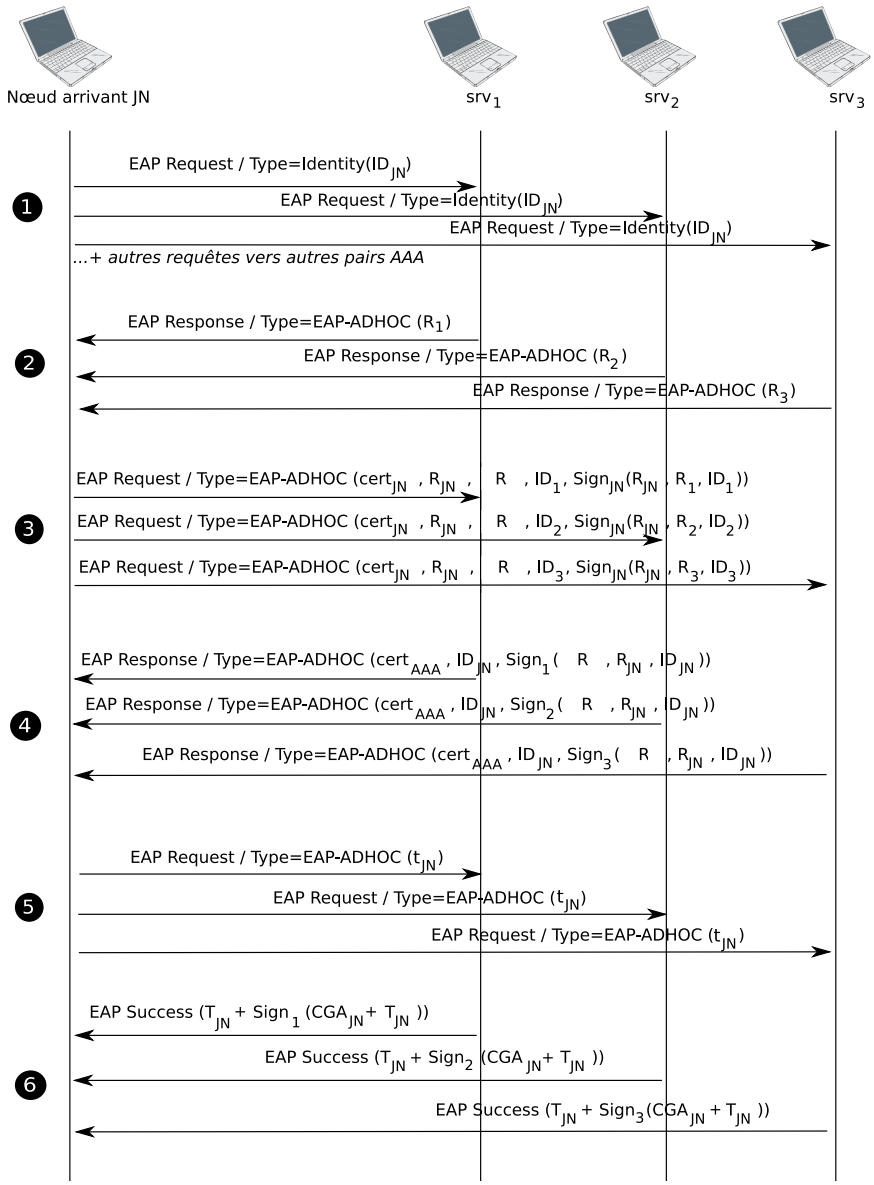


FIGURE 4.4 – Protocole d'authentification distribué

premier à envoyer un défi aux pairs AAA, elle l'oblige à être le premier à effectuer les opérations cryptographiques. Ainsi, on évite l'attaque par déni de service visant à épuiser les ressources en calculs et/ou en mémoire des pairs.

**Étape 2** On appelle $\mathcal{P}_2$ l'ensemble des pairs ayant reçus les requêtes du JN ($\mathcal{P}_2 \subseteq \mathcal{P}_1 \subseteq \mathcal{P}$). Chaque pair $\text{srv}_\text{j}$ de $\mathcal{P}_2$ vérifie l'existence du JN dans sa base des utilisateurs. Si l'identifiant du JN n'y figure pas, il envoie un message EAP de type échec c.-à-d. un message EAP Failure. Sinon, il répond par un

message EAP de type EAP-ADHOC demandant au JN de s'authentifier en utilisant la méthode d'authentification EAP-ADHOC.

En suivant les spécifications du protocole ISO 9798-3 (cf. section 4.4.1), ce message doit contenir un défi, $R_j$, que signera le JN, à l'étape 3, en utilisant sa clé privée. La validité de la signature doit pouvoir être vérifiée par $srv_j$ à l'étape 4. Si la vérification réussit, $srv_j$ doit co-signer avec d'autres pairs de $\mathcal{P}_2$ le défi $R_j$ ainsi que des informations supplémentaires. En conséquence, $R_j$ doit être connu de ces autres pairs. Ce qui *a priori* n'est pas le cas, d'où la nécessité de modifier d'avantage le protocole ISO 9798-3, dès l'étape 2, afin de résoudre ce problème.

On peut proposer plusieurs solutions : la méthode du délégué tournant ($m1$), la méthode de l'information préalable ($m2$), la méthode des nombres aléatoires distincts ($m3$).

($m1$) *Méthode du délégué tournant :* dans cette solution, le groupe des pairs va choisir l'un d'entre eux, que nous appellerons le *délégué*, pour poser le défi au nom du groupe. Par convention, le premier délégué est le pair $srv_1$. Le pair $srv_i$ choisi tire au hasard un nombre R et l'envoie à chacun des autres pairs de $\mathcal{P}_I$. Puis, chacun des pairs envoient R à JN (sur la figure 4.4, $R_1 = R_2 = R_3 = R$). Le délégué $srv_j$ qui agira au nom du groupe à la prochaine demande sera obtenu en calculant j = R mod $n$. Pour cela, le pair $srv_j$ envoie un message d'acquittement à l'ensemble des pairs. Si le pair $srv_{j+1}$ n'a pas reçu ce message, il sait que le pair $srv_j$ n'est pas en mesure d'assurer sa tâche et prend le relais en envoyant à son tour un message d'acquittement. Le processus se répète ainsi jusqu'à ce que l'une des deux conditions soit satisfaite : soit on a trouvé le délégué pour la prochaine émission, soit aucun des pairs ne peut agir et le réseau est en panne. Une rapide évaluation montre que, dans des conditions normales, au plus $3\,n - 2$ messages sont échangés. Les différents pairs n'étant pas nécessairement munis du même générateur, un avantage de cette méthode est la robustesse du tirage aléatoire qui permet aux pairs AAA de connaître tous le même nombre R. L'inconvénient majeur reste une certaine sensibilité à la perte de messages qui rend l'utilisation de cette méthode dangereuse lorsque la probabilité de perte de messages s'élève, par exemple pour un MANET dispersé et très mobile.

($m2$) *Méthode de l'information préalable :* synchroniser les différents pairs afin qu'ils puissent tous générer le même défi R pose de grande difficulté. Dans cette méthode, nous allons utiliser une information préalablement synchronisée dans l'ensemble des pairs, à savoir l'heure. A chaque requête d'un JN, chacun des pairs initialise son générateur aléatoire avec le nombre de secondes écoulées depuis une date conventionnelle. Il est nécessaire que chaque pair dispose du même générateur et connaisse préalablement la date conventionnelle. Dans ces conditions, chacun des générateurs de chaque pair donne un même nombre R qui est utilisé par la suite (sur la figure 4.4, $R_1$=$R_2$=$R_3$=R). Le premier inconvénient de cette méthode est le manque de robustesse. Une seconde difficulté plus subtile réside dans le fait que deux requêtes à des pairs distincts peuvent ne pas arriver dans la même seconde. Elle peut être résolue en comptant des paires ou des di-

zaines de secondes mais alors la robustesse de la génération aléatoire s'en trouve affectée. Notons toutefois que l'algorithme à tentatives multiples décrit dans la section 4.4.5 limite cet inconvénient.

($m3$) *Méthode des nombres aléatoires distincts :* dans cette solution, les messages issus de chaque pair de $\mathcal{P}_2$ comprennent un nombre aléatoire court (par exemple deux ou trois octets) mais non nécessairement égaux, soient $\mathrm{R}_{\mathrm{i}_1}$, $\mathrm{R}_{\mathrm{i}_2}$,$\cdots$, $\mathrm{R}_{\mathrm{i}_\gamma}$ où $\gamma = \mathrm{card}(\mathcal{P}_2)$.A l'étape 3, JN utilisera ces nombres aléatoires pour construire un nombre noté R qui sera envoyé aux pairs AAA. Ainsi, chaque pair pourra, en consultant la tranche qui le concerne de ce nombre, s'assurer qu'au moins en ce qui le concerne il n'y a pas eu de falsification. Il pourra vérifier, par la suite, la signature du JN. De plus, grâce à R, les pairs pourront co-signer les mêmes informations à l'étape 4.

Nous avons opté pour la méthode $m3$ qui est plus facile à réaliser que les deux autres.

**Étape 3** Si JN reçoit des réponses différentes d'EAP Failure et dont le nombre est supérieur ou égal à $th$, JN peut poursuivre les étapes du protocole. Sinon, il ne le peut pas et l'authentification échoue.

Dans le cas positif, on appelle l'ensemble des pairs ayant répondu $\mathcal{P}_3$. On a alors : $\mathcal{P}_\kappa \subseteq \mathcal{P}_3 \subseteq \mathcal{P}_2 \subseteq \mathcal{P}_1 \subseteq \mathcal{P}$. JN prépare un deuxième type de requête pour au moins $th$ pairs choisis dans $\mathcal{P}_3$, par exemple ceux qui étaient les plus rapides à répondre. Soit $\mathcal{P}_4$ leur ensemble, on a alors $\mathcal{P}_\kappa \subseteq \mathcal{P}_4 \subseteq \mathcal{P}_3 \subseteq \mathcal{P}_2 \subseteq \mathcal{P}_1 \subseteq \mathcal{P}$.

On définit la fonction [12] $\sigma$ qui, à chaque famille de nombres aléatoires issue d'un ensemble $\mathcal{Q}$ (de cardinal $q$) de pairs, associe :

$$\sigma(\mathcal{Q}) = (\mathrm{ID}_{\mathrm{i}_1} + \mathrm{R}_{\mathrm{i}_1}) + (\mathrm{ID}_{\mathrm{i}_2} + \mathrm{R}_{\mathrm{i}_2}) + \cdots + (\mathrm{ID}_{\mathrm{i}_q} + \mathrm{R}_{\mathrm{i}_q})$$

La concaténation, notée « + », étant étendue à tous les pairs $\mathrm{srv}_{\mathrm{i}_\mathrm{j}}$ de $\mathcal{Q}$ et les $(\mathrm{ID}_{\mathrm{i}_\mathrm{j}})_{1 \leqslant j \leqslant q}$ étant les identités de ces mêmes pairs. Ainsi $\sigma(\mathcal{Q})$ est la concaténation des identités et des nombres aléatoires des pairs de l'ensemble $\mathcal{Q}$.

En utilisant la fonction $\sigma$ définie ci-dessus JN calcule, d'abord, $\mathrm{R} = \sigma(\mathcal{P}_4)$. Il choisit, ensuite, un nombre aléatoire $\mathrm{R}_{\mathrm{JN}}$ et signe le triplet ($\mathrm{R}_{\mathrm{JN}}$, $\mathrm{R}_{\mathrm{i}_\mathrm{j}}$, $\mathrm{ID}_{\mathrm{i}_\mathrm{j}}$) pour chaque pair $\mathrm{srv}_{\mathrm{i}_\mathrm{j}} \in \mathcal{P}_4$.

Puis, chaque requête destinée à un pair $\mathrm{srv}_{\mathrm{i}_\mathrm{j}} \in \mathcal{P}_4$ est construite comme un message EAP de type EAP-ADHOC contenant : le certificat du JN c.-à-d. $\mathrm{cert}_{\mathrm{JN}}$, le nombre aléatoire $\mathrm{R}_{\mathrm{JN}}$, le nombre R, l'identité de $\mathrm{srv}_{\mathrm{i}_\mathrm{j}}$ c.-à-d. $\mathrm{ID}_{\mathrm{i}_\mathrm{j}}$ et la signature du JN c.-à-d. $\mathrm{sign}_{\mathrm{JN}}(\mathrm{R}_{\mathrm{JN}}, \mathrm{R}_{\mathrm{i}_\mathrm{j}}, \mathrm{ID}_{\mathrm{i}_\mathrm{j}})$.

Enfin, les requêtes sont envoyées à l'ensemble $\mathcal{P}_4$.

**Étape 4** On appelle $\mathcal{P}_5$ l'ensemble des pairs ayant reçus les requêtes du JN ($\mathcal{P}_5 \subseteq \mathcal{P}_4 \subseteq \mathcal{P}_3 \subseteq \mathcal{P}_2 \subseteq \mathcal{P}_1 \subseteq \mathcal{P}$). Chacun des pairs de $\mathcal{P}_5$ vérifie, dans la requête qui lui a été adressée, d'abord l'existence de son nombre aléatoire dans la bonne tranche de R ensuite la validité de la signature (cf. section B.5.2 de l'annexe B) reçue. Si ces conditions n'ont pas été établies, il envoie un

12. Du grec *συνενώνει*, je concatène.

message EAP Failure. Sinon, il répond par un message EAP de type EAP-ADHOC contenant le certificat commun aux pairs AAA c.-à-d. $\mathrm{cert}_{\mathrm{AAA}}$, l'identité du JN c.-à-d. $\mathrm{ID}_{\mathrm{JN}}$ et sa signature partielle (cf. section 4.4.2.2) sur (R, $\mathrm{R}_{\mathrm{JN}}$, $\mathrm{ID}_{\mathrm{JN}}$).

**Étape 5** Si JN ne reçoit pas au moins *th* réponses différentes de EAP Failure, l'authentification échoue. Sinon, le JN peut poursuivre en procédant à la reconstitution de la signature *globale*[13] de l'ensemble des pairs AAA (cf. section 4.4.2.2) noté $\mathcal{P}_6$ ($\mathcal{P}_\kappa \subseteq \mathcal{P}_6 \subseteq \mathcal{P}_5 \subseteq \mathcal{P}_4 \subseteq \mathcal{P}_3 \subseteq \mathcal{P}_2 \subseteq \mathcal{P}_1 \subseteq \mathcal{P}$). Ensuite, à l'aide de la clé publique contenue dans le certificat $\mathrm{cert}_{\mathrm{AAA}}$, il vérifie sa validité. Si cette signature n'est pas valide, l'authentification échoue. Sinon, l'authentification de l'ensemble des pairs AAA réussit et par conséquent l'authentification mutuelle aussi. JN informe les pairs de $\mathcal{P}_7$ ($\mathcal{P}_\kappa \subseteq \mathcal{P}_7 \subseteq \mathcal{P}_6$) de cette réussite en envoyant à chacun un message EAP de type EAP-ADHOC contenant la date $\mathrm{t}_{\mathrm{JN}}$ de cet envoi. Cela signifie que le JN demande à être autorisé aux services souscrits à dater de cet instant.

**Étape 6** Chaque pair ayant reçu le message du JN contenant $\mathrm{t}_{\mathrm{JN}}$ vérifie la plausibilité de cette date au regard de son horloge interne[14]. Si la vérification par un pair $\mathrm{srv}_{\mathrm{i_j}}$ échoue, le JN est considéré comme un nœud malicieux et un message EAP Failure est envoyé au JN. Sinon, le pair répond par un message EAP Success qui contient un morceau du jeton d'accès du JN précisant la date limite, $\mathrm{T}_{\mathrm{JN}}$, de ses droits. Au delà de cette date, JN doit se ré-authentifier. La génération de morceaux d'un jeton est identique à celle de morceaux d'une signature partagée RSA. Donc, le JN doit récolter au moins *th* morceaux. Chaque morceau est calculé en utilisant $\mathrm{T}_{\mathrm{JN}}$ et $\mathrm{CGA}_{\mathrm{JN}}$ qui est l'adresse IP du JN, générée de manière cryptographique (cf. section C.3 de l'annexe C).

Nous présentons des éléments plus approfondis concernant la génération et l'utilisation d'un jeton d'accès dans la section 4.5. Nous y expliquerons en particulier le choix des adresses IP générées de manière cryptographique.

La cinquième et la sixième étapes ne figurent pas dans les spécifications de la norme ISO 9798-3 (cf. section 4.4.1) et servent à préparer l'étape d'autorisation.

Notons que, même si la figure 4.4 présentent les différentes étapes comme étant séparées, dans un réseau réel cela n'est pas forcément toujours le cas. En effet, certaines requêtes peuvent voir leurs réponses envoyer avant que certaines autres requêtes n'arrivent à destination.

Concernant les procédures d'authentification échouées, un client peut relancer une nouvelle tentative d'authentification selon un algorithme décrit à la section 4.4.5.

13. On entend par signature *globale*, la signature RSA obtenue après combinaison des parts de signature envoyées par les pairs AAA. La signature globale, une fois reconstituée, apparaît comme une signature RSA générée par une seule entité.

14. Selon la taille du réseau c.-à-d. le nombre de nœuds dans le réseau et selon l'écart d'horloge toléré, on peut, par exemple, admettre une différence de 1 à 5 secondes.

### 4.4.4 Optimisation du nombre d'étapes

Nous nous intéressons dans cette sous section à l'optimisation du nombre de messages AAA d'authentification circulant dans le réseau. Ce travail a fait l'objet d'une publication à MWNS, un atelier qui a eu lieu dans le cadre de la conférence IFIP Networking en 2009 [83]. Bien que la proposition ci-dessus reste assez fidèle à l'esprit du protocole d'authentification mutuelle ISO 9798-3, elle introduit trois nouveaux types de messages, soit trois étapes, supplémentaires par rapport au protocole initial. Comme une étape consiste à envoyer non pas un seul message mais plusieurs, *le nombre de messages par étape est au moins égal à $th$ et au plus égal à $n$.*. Le nombre de messages pour accomplir une authentification se trouve fortement augmenté. Il est donc utile d'optimiser le nombre de messages.

Deux approches peuvent être considérées. La première consiste à diminuer le nombre d'étapes du protocole. Cela implique la modification du protocole même ou du moins la modification de la structure de certains de ses messages. La deuxième approche consiste à diminuer le nombre de messages envoyés par étape. Cela concerne la définition du nombre de pairs ciblés à chaque nouvel envoi et est en particulier lié aux valeurs des paramètres $n$ et $th$.

Le problème d'optimisation ainsi décrit est en quelque sorte à trois dimensions qui se résument à : nombre d'étapes du protocole, nombre de pairs $n$ et seuil cryptographique $th$. Alors que la première approche peut être réalisée par une simple étude du contenu des messages, la deuxième nécessite une étude de performances, en terme de disponibilité et de sécurité. Comme il s'agit dans ce chapitre de la définition du protocole et non de l'étude de ses performances, nous optons dans un premier temps pour la modification du protocole. Une analyse de performances sera donnée au chapitre 7.

On remarque d'abord, qu'en réalité, il suffit que le JN soit authentifié pour lui délivrer son jeton d'accès. Il est ensuite libre de l'utiliser ou de le supprimer s'il ne réussit pas à authentifier l'ensemble des pairs AAA. De plus, si à la dernière étape le nombre de morceaux du jeton reçus par le JN n'avaient pas été suffisants pour le constituer, les étapes précédentes n'auraient au bout du compte contribué qu'au gaspillage de la bande passante et des ressources des pairs AAA. Autant, alors, délivrer les morceaux du jeton en même temps que les morceaux de la signature des pairs. Cela nécessite que l'envoi par JN de la date $t_{JN}$, prévu à l'étape 5, soit avancé à l'étape 3.

Ainsi, il est possible de condenser les quatrième, cinquième et sixième étapes en une seule étape où chaque message transporte à la fois les éléments d'authentification des pairs AAA et les parts du jeton du JN. En revanche, il n'est pas pratique d'opérer davantage de réduction du nombre d'étapes car le premier type de message a été pensé pour éviter une attaque de déni de services par consommation de ressources, le deuxième et le troisième permettent une authentification du JN par question/réponse, le troisième et le quatrième permettent une authentification par question/réponse du service AAA, enfin le quatrième transporte, entre autres, le jeton du JN.

Les protocoles AAA tels que RADIUS et Diameter supportent l'extension par, en particulier, la possibilité de définir de nouveaux pairs d'attribut et de valeur (en anglais *Attribute Value Pair* ou AVP) pour les nouveaux besoins des applications [15]. Un AVP est un champ contenant un couple (nom d'attribut,

15. Des applications telles que Mobile IP utilise les protocoles AAA pour assurer l'authen-

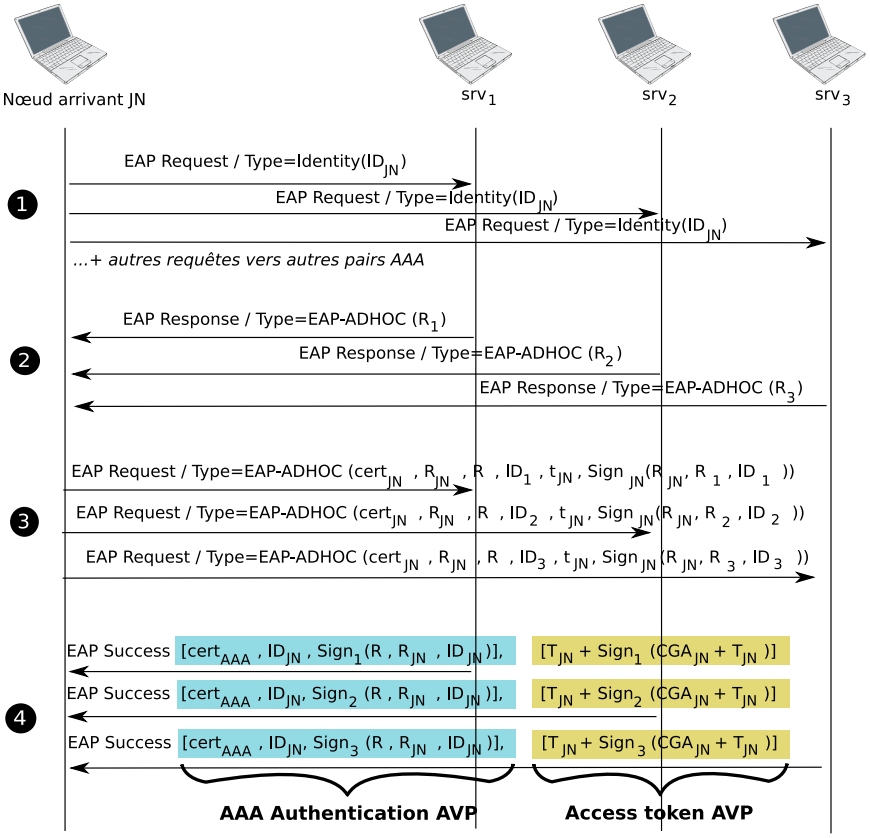


FIGURE 4.5 – Optimisation du protocole proposé

valeur) permettant le transport de données spécifiques concernant l'authentification, l'autorisation, la comptabilité, le routage ou la sécurité. A sa réception, s'il n'est pas interprétable par un équipement AAA, il est alors ignoré.

Nous définissons donc deux nouveaux AVP : un AVP d'authentification du service AAA (en anglais *AAA Authentication AVP*) transportant les éléments d'authentification des pairs AAA et un AVP de jeton d'accès (en anglais *Access Token AVP*) permettant le transport des parts d'un jeton. La figure 4.5 illustre ces AVP et leur utilisation.

### 4.4.5 Algorithme à multiples tentatives d'authentification

Trois raisons peuvent provoquer l'échec d'une authentification :

1. aucune réponse ne parvient au JN lors du premier ou du second échange,
2. seulement des messages EAP Failure sont reçus au premier ou au deuxième échange,
3. enfin, des réponses EAP autres que EAP Failure sont reçues au premier ou au deuxième échange mais leur nombre est inférieur à *th* donc insuffisant pour pouvoir poursuivre l'authentification.

tification, l'autorisation et la comptabilité dans des conditions d'accès particulières.

Lorsque l'une de ces situations se présente, JN doit être en mesure de lancer de nouvelles tentatives d'authentification. Le temps entre deux tentatives successives et le nombre de tentatives maximal sont deux paramètres dont les valeurs sont fixées à l'avance ou calculées selon un algorithme déterminé.

Nous avons proposé un algorithme de renouvellement d'authentification en cas d'échec, appelé algorithme à tentatives multiples d'authentification (en anglais *Multiple-attempt back-off algorithm*), à l'occasion d'une publication à la conférence IEEE ISCC 2011 [84]. Un tel algorithme utilise trois paramètres : la durée maximale d'attente avant d'abandonner une tentative d'authentification, la durée maximale d'attente avant d'en initier une nouvelle et le nombre maximum de tentatives. Pour simplifier, nous supposons que la durée maximale d'attente avant d'abonner une tentative et celle avant de renouveler la demande sont identiques. Cette durée, initialement égale à $t_{max}$, est doublée après chaque échec. Le nombre maximal de tentatives est noté $K$. Nous avons déterminé les valeurs de $t_{max}$ et de $K$ grâce à la technique de simulation dont les résultats sont présentés à la section 7.6 du chapitre 7.

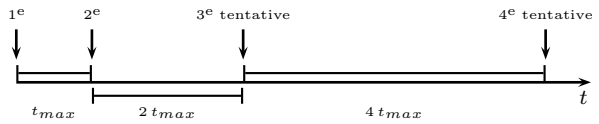


FIGURE 4.6 – Algorithme à multiples tentatives d'authentification

La figure 4.6 décrit le déroulement de l'algorithme à multiples tentatives : si la première tentative échoue après une attente de $t_{max}$, JN initie une nouvelle tentative. Si celle-ci échoue après une attente de durée $2\,t_{max}$, JN en essaye une troisième. De manière générale, après avoir lancé une $j^{\text{ème}}$ tentative, JN attend un intervalle de temps de durée $2^{j-1}\,t_{max}$ avant d'en essayer une nouvelle. Si après $j$ tentatives, le processus d'authentification se termine avec succès, l'intervalle total attendu aura été entre $(2^{j-1}-1)\,t_{max}$ et $(2^j-1)\,t_{max}$. Enfin, après $K$ tentatives non réussies, le JN abandonne l'authentification. Ce mécanisme de *back-off* permet d'éviter les mauvaises conditions de transmission dans le réseau en espaçant les tentatives d'un intervalle de temps dont la longueur croit exponentiellement. Il permet ainsi d'augmenter les chances de réussite d'une authentification.

Fred,
*Le Naufragé du « A » :*
découverte du deuxième A

## 4.5 Protocoles d'autorisation proposés

A l'issue de l'authentification mutuelle entre un nœud client et un ensemble de pairs, le client récupère un jeton d'accès (en anglais *Access Token* ou AT) de durée limitée dont la date limite, notée T. Lorsque cette date expire, le client doit se ré-authentifier afin d'acquérir un autre jeton. L'autorisation à tout service repose nécessairement sur la présentation d'un jeton valide qui permet, entre autres, au client de faire valoir ses droits.

La section 3.3.3.1 du chapitre 3 montre que plusieurs services différents peuvent être fournis dans un MANET. Nous nous sommes intéressés, dans cette thèse, au service de routage et à celui de contenus. Dans chacun des cas, les parties impliquées sont différentes, c'est pourquoi il est nécessaire de concevoir un protocole d'autorisation différent pour chaque service. Chacun d'entre eux fait usage du jeton d'accès comme preuve de l'autorisation du client à ces services.

### 4.5.1 Jeton d'accès

Afin de pouvoir lier l'identité d'un client à son jeton et d'empêcher ainsi son vol, nous faisons appel aux techniques CGA et SEND, toutes les deux décrites dans la section C.3 de l'annexe C. Pour cela, nous supposons que le client a préalablement calculé son adresse IPv6 comme une adresse CGA qui est donc liée par des moyens cryptographiques à sa clé publique. Une signature électronique générée par les pairs sur la concaténation de T avec cette adresse permet ensuite de garantir la protection du jeton contre la modification. Comme pour l'authentification, la génération de cette signature et, par conséquent, celle du jeton est effectuée de façon partagée entre les pairs, d'où notre proposition suivante de l'expression du jeton :

$$\mathrm{AT}=\mathrm{T} + \mathrm{sign}_{\mathrm{AAA}}(\mathrm{CGA} + \mathrm{T})$$

Le symbole $\mathrm{sign}_{\mathrm{AAA}}$ est celui de la signature de groupe des pairs AAA (cf. tableau 4.2).

Les étapes d'acquisition d'un jeton sont les suivantes :

1. chaque pair $\mathrm{srv}_{\mathrm{j}}$ sollicité par un message envoyé à la troisième étape calcule le temps limite T en ajoutant à $t$, la date d'envoi de ce message, la durée

de validité d'un jeton, par exemple une heure. Il calcule ensuite le morceau du jeton du client $AT_j$=T + $sign_j$(CGA + T).

2. $srv_j$ envoie $AT_j$ à JN (cf. FIGURE 4.5),
3. JN reçoit les morceaux de son jeton et n'exécute l'étape 4 que s'il en a reçu au moins *th*,
4. JN sépare, dans chaque $AT_j$, la partie gauche correspondant au temps limite T de la partie droite correspondant au morceau de signature $sign_j$(CGA + T),
5. JN combine les morceaux de signature et obtient $sign_{AAA}$(CGA + T) qu'il concatène avec T pour obtenir le jeton AT=T + $sign_{AAA}$(CGA + T).

#### 4.5.1.1 Considérations de sécurité

Avant toute utilisation, un jeton doit être construit. Cette construction met en jeu trois parties : le client, les pairs et les nœuds relais. Le client peut-il présenter une fausse CGA ? Non parce que, par construction, SEND garantit que les nœuds voisins du client ne relayent pas son trafic sans avoir auparavant exécuté correctement SEND. Ainsi les nœuds voisins ont vérifié la validité de l'adresse du client. Les pairs pourraient-ils modifier cette adresse ? Non parce que cela nécessiterait un nombre de pairs corrompus supérieur au seuil pour générer un nouveau jeton et que cette hypothèse est hautement improbable en raison du rafraîchissement des clés. Enfin, un nœud relais peut-il effectuer cette modification ? C'est en théorie possible si ce nœud est un point d'articulation du graphe auquel cas il lui est possible d'intercepter tous les messages du client d'une part et tous ceux des pairs d'autre part. On voit que si la sécurité n'est pas garantie, les conditions exceptionnelles nécessaires à la mise en défaut du protocole sont extrêmement rares, comme est extrêmement improbable le fait pour un nœud se trouvant dans les conditions nécessaires de connaître qu'il est effectivement dans cette situation.

Néanmoins, il est possible de modifier une partie du jeton correspondant à la date limite T, et cela à des fins d'empêcher le JN de pouvoir accéder à tout service. C'est une attaque de déni de service qui est à priori improbable si l'on met en place, dans le voisinage de chaque nœud, des techniques de surveillance du trafic qu'il émet [16].

### 4.5.2 Autorisation au service de routage

L'autorisation au service de routage est effectuée dans le voisinage par les nœuds voisins qui ont déjà été authentifiés et autorisés. Elle se fait grâce aux deux mécanismes suivants :

1. l'exécution du protocole SEND dans le voisinage,
2. l'adjonction du jeton d'accès, reçu à la fin d'une authentification terminée avec succès, à chaque paquet émis

En effet, un nœud arrivant JN, ayant été authentifié, voit la validité de son jeton d'accès vérifiée avant toute opération de relais c.-à-d. de routage. La vérification par un nœud voisin (en anglais *Neighbor Node* ou NN) fait appel aux éléments qui se trouvent dans sa base des utilisateurs autorisés (cf. section 4.3.2.2) et dans

16. Une de ces techniques est la technique de *watchdog*.

son cache des voisins (cf. section C.3.1 de l'annexe C) et au certificat du service AAA enregistré à l'étape d'inscription (cf. section 4.3.2.2). Elle se déroule selon les étapes décrites par l'organigramme de la figure 4.7.

On note $M_r$ un message reçu par NN de la part du JN, $CGA_r$ son adresse source et $AT_r$ le jeton qu'il contient. On note aussi $T_e$ et $AT_e$ respectivement la date d'expiration du jeton du JN et sa valeur dans la base des utilisateurs du NN quand une entrée pour JN y existe. Enfin, on note $CGA_e$ l'adresse CGA du JN quand une entrée pour JN existe dans le cache des voisins du NN. Soit $t$ la date à laquelle $M_r$ est reçu, alors l'algorithme de vérification de la validité du $AT_r$ suit l'enchaînement suivant :

1. NN extrait $AT_r$ du $M_r$ et $CGA_r$ de l'en-tête IP de celui-ci,
2. NN vérifie s'il a déjà une entrée pour JN dans sa base des utilisateurs autorisés. Si une telle entrée existe, NN passe à l'étape 3, sinon il passe à l'étape 4.
3. NN extrait $T_e$ et $AT_e$ de l'entrée trouvée et vérifie que $T_e$ n'a pas expiré et que $AT_r$ est bien égal à $AT_e$ pour router $M_r$. Si tel n'est pas le cas, trois possibilités se présentent au NN :
   - $T_e$ a expiré et $AT_r$=$AT_e$ : JN n'est plus autorisé au routage et $M_r$ est jeté,
   - $T_e$ a expiré et $AT_r \neq AT_e$ : l'ancien jeton du JN n'est plus valide et il a un nouveau jeton dont NN doit vérifier la validité en allant à l'étape 4.
   - $T_e$ n'a pas expiré mais $AT_r \neq AT_e$ : l'ancien jeton du JN n'est plus valide car JN a demandé un nouveau jeton aux pairs AAA après qu'il soit authentifié à nouveau [17]. NN doit vérifier la validité du nouveau jeton en allant à l'étape 4.
4. NN vérifie s'il a déjà une entrée pour JN dans son cache des voisins. Si une telle entrée existe [18], NN extrait $CGA_e$ de l'entrée trouvée et vérifie si $CGA_r$ est égale à $CGA_e$ [19] : si cette égalité est établie, il va à l'étape 6, sinon, il jette $M_r$. S'il n'existe pas d'entrée pour JN, NN va à l'étape 5.
5. NN exécute SEND (cf. l'annexe C) avec JN et récupère l'adresse CGA de celui-ci afin de la vérifier. Il déroule pour cela l'algorithme de vérification de validité d'une adresse CGA, présenté à la section C.3.1 de l'annexe C. Si cette validité est établie, NN crée une entrée pour JN dans son cache des voisins et va à l'étape 6, sinon $M_r$ est jeté.
6. Extraire $T_r$ de $AT_r$ en prenant la partie à gauche de la signature. Si $t \leq T_r$, passer à l'étape 7, sinon jeter $M_r$.
7. Extraire la signature $sign_{AAA}\{CGA + T\}$ et vérifier qu'elle est valide comme décrit à la section B.5.2 de l'annexe B. Si sa validité est établie, mettre à jour la base des utilisateurs autorisés ensuite router $M_r$, sinon le jeter. La mise à jour est effectuée par la création d'une entrée pour JN si elle n'existe pas, sinon par sa modification si elle existe déjà.

17. Cela peut arriver si JN a perdu son jeton suite à un redémarrage ou s'il a soupçonné que son jeton a été volé.
18. Une entrée pour JN existe s'il était déjà dans le voisinage de NN et que les deux nœuds ont exécuté SEND.
19. Ce test est suffisant car si une CGA existe déjà dans le cache, cela veut dire que sa validité a été préalablement établie à l'exécution de SEND en supposant qu'une corruption de la base de données n'a pas eu lieu.

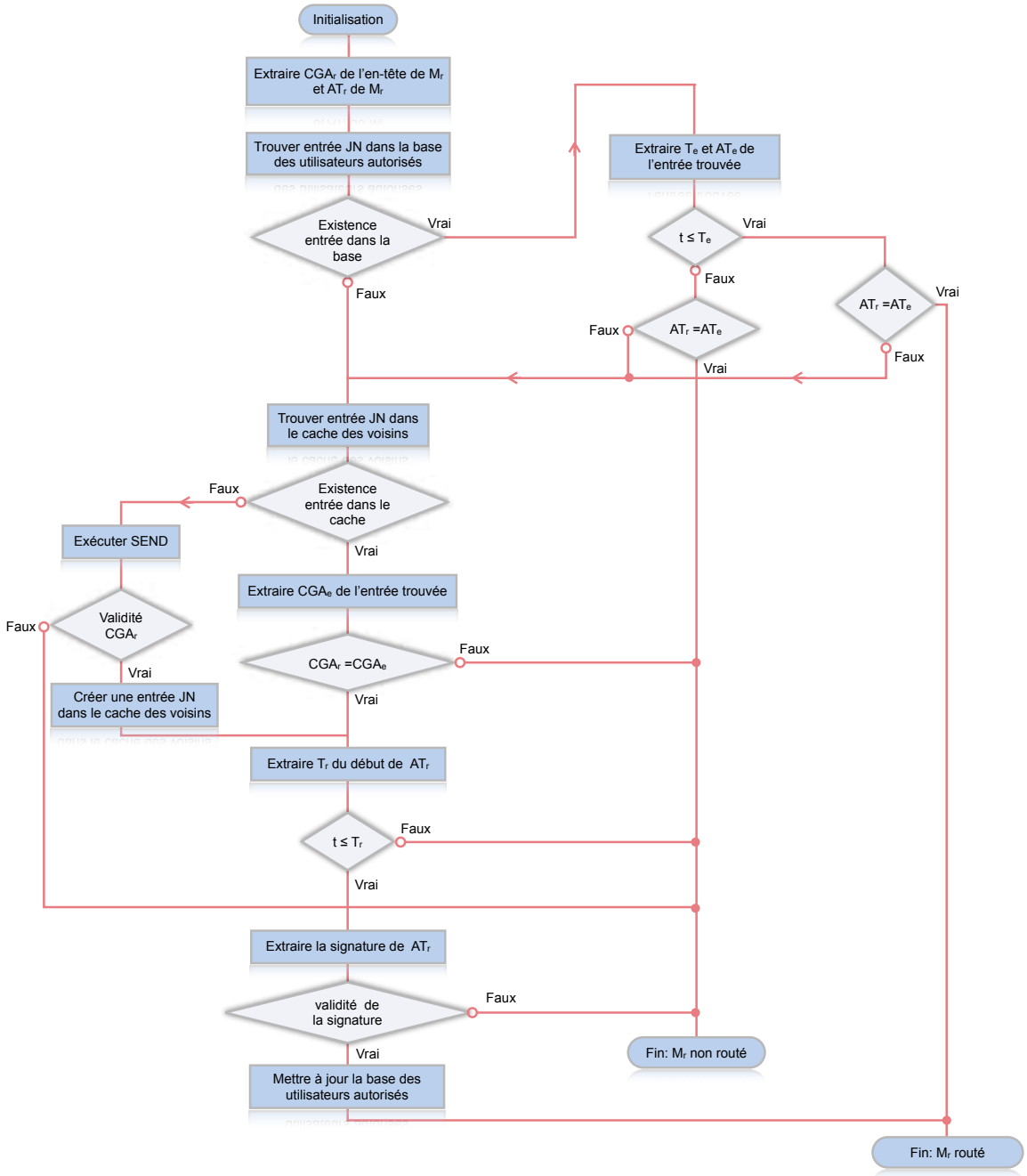


FIGURE 4.7 – Algorithme de vérification de la validité d'un jeton d'accès

Il est à remarquer que les étapes 1 à 3 évitent au NN de vérifier "entièrement" la validité du jeton en validant la valeur de la signature à chaque réception d'un message du JN. Celle-ci ne devient nécessaire que si c'était la première fois qu'un message était reçu du JN ou dans des cas particuliers comme expliqué ci-dessus.

Par ailleurs, il est possible que NN ne connaisse pas JN mais qu'il reçoive de sa part un message SEND contenant son jeton parce qu'il avait déjà effectué l'étape d'authentification alors qu'il n'était pas encore voisin de NN. Dans ce cas, JN ne sollicite pas NN pour un routage, mais ce dernier doit procéder à la vérification de la CGA du JN ensuite à celle du jeton ainsi que nous l'avons décrit auparavant.

### 4.5.3 Autorisation au service de contenus

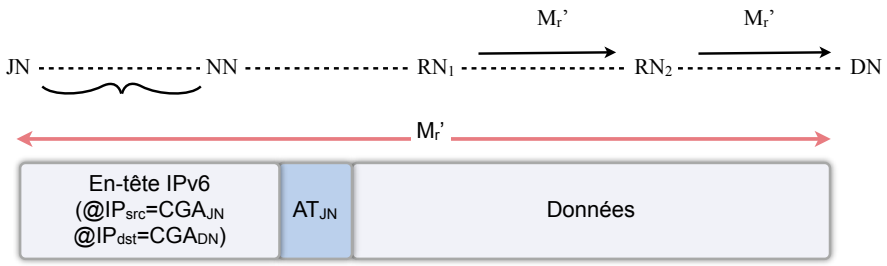


FIGURE 4.8 – Attaque par vol du jeton

Pour faire valoir son droit au service de contenus fourni par un nœud distributeur (en anglais *Distributor node* ou DN), un nœud client JN envoie son jeton. Mais cela ne suffit pas pour qu'un DN soit assuré de l'authenticité de l'identité du JN et de celle de son jeton. En effet, il serait facile pour un nœud intermédiaire, se trouvant sur le chemin entre JN et DN d'intercepter les messages du JN, de voler son jeton et de s'en servir après. La figure 4.8 illustre cela en montrant un nœud relais (en anglais *Relay node* ou RN) générant un message $M_r$' à destination de DN, ayant comme adresse source la CGA du JN et contenant le jeton de celui-ci. Comme DN n'est pas un voisin de JN, il n'a pas précédemment reçu les messages SEND de celui-ci et n'a donc pas les moyens de vérifier la validité de l'adresse source et la validité du jeton. Il les accepte donc tels quels et l'attaque par vol du jeton réussit. Pour cette raison, nous proposons d'étendre l'utilisation de SEND au delà du voisinage immédiat [20] d'un nœud chaque fois que cela est nécessaire, par exemple dans le cas qui nous intéresse c.-à-d. l'accès à un service de contenus.

La figure 4.9 montre le format d'un message de SEND étendu (en anglais *Extended SEND* ou Ext-SEND). Ce protocole est à deux messages et est utilisé avant toute autorisation au service de contenus par DN entre JN et ce dernier. En plus des options SEND habituelles, les messages Ext-SEND transporte le jeton du JN et, accessoirement, des paramètres Diffie-Hellman [85]. L'option RSA est, par ailleurs, calculée en prenant en compte ces nouveaux éléments et n'est ajoutée qu'à la suite au message Ext-SEND. Ainsi, non seulement le vol d'adresses et le vol de jeton sont évités mais aussi l'attaque de l'homme du milieu pendant l'échange des paramètres Diffie-Hellman est contrée [21]. Le

20. Le voisinage immédiat d'un nœud JN est composé par les nœuds atteints en un seul saut par ses message.

21. L'attaque de l'homme du milieu est possible pendant les échanges Diffie-Hellman. Eve

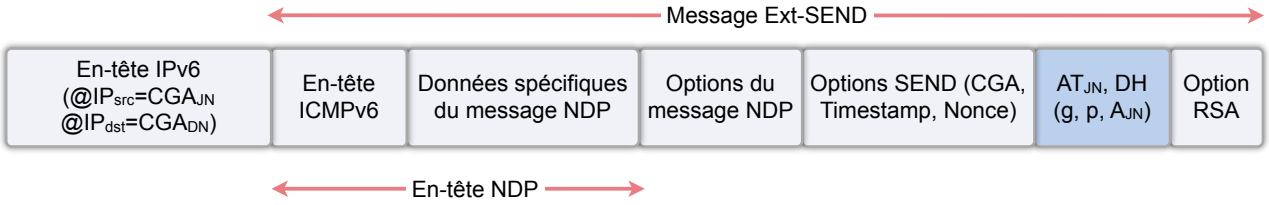


FIGURE 4.9 – Format d'un message SEND étendu (Ext-SEND)

but de l'utilisation optionnelle de la technique Diffie-Hellman est de permettre à JN et DN de construire une clé partagée secrète afin de pouvoir assurer la confidentialité de leurs échanges ultérieurs (cf. FIGURE 4.9).

Quand DN reçoit un message Ext-SEND, il procède à la validation de la CGA et du jeton du JN comme nous l'avons décrit à la section 4.5.2. Il vérifie aussi la validité de la signature du JN se trouvant dans l'option RSA. Cette validité apporte la garantie de l'authenticité du JN. Quand ces tests se révèlent positifs, DN calcule la clé Diffie-Hellman qu'il pourra partager avec JN. Il récupère, pour cela les paramètres $g$ et $p$, choisis par JN, génère une clé secrète $b$, et calcule $\mathrm{A_{DN}}=g^b \ mod \ p$. Remarquons que de la même manière, JN avait auparavant choisi une clé secrète $a$ et calculé $\mathrm{A_{JN}}=g^a$ mod $p$. La clé partagée est alors :

$$\mathrm{K}=\mathrm{A_{JN}}^b \ mod \ p=\mathrm{A_{DN}}^a \ mod \ p$$

DN répond, ensuite, à JN par un message Ext-SEND ayant le même format que que celui reçu, mais contenant sa propre CGA, son jeton, son option RSA ainsi que d'autres paramètres de ses options SEND.

JN opère les mêmes vérifications sur les éléments du DN à la réception de sa réponse. Si elles ont toutes abouti sur un succès, JN et DN auront achevé une authentification mutuelle réussie et auront aussi établi une clé partagée. De plus, JN aura justifié, grâce à son jeton, de son autorisation aux services de contenus et DN, grâce au sien, de son autorisation à fournir ce contenu.

peut choisir un paramètre $\mathrm{E}_A$ pour mettre en place une clé partagée avec A et une autre clé $\mathrm{E}_B$ pour les échanges avec B en utilisant le paramètre $g$. Ni A, ni B ne pourront détecter cette attaque car ils ne se sont pas authentifiés mutuellement. L'option RSA, présente déjà dans les messages SEND, permet cette authentification.

Fred,
*Le Naufragé du « A » :*
découverte du troisième A

### 4.5.4 Collectes d'informations pour les besoins de comptabilité

Chaque fois qu'un pair délivre un jeton à un client, il enregistre dans sa base de données de comptabilité la durée de ce jeton dans l'entrée correspondante à ce client. Les pairs envoient régulièrement, quand ils sont connectés à un réseau de FAI, un compte rendu de cette collecte à un serveur AAA central afin qu'il établisse une comptabilité détaillée et puisse plus tard émettre une facturation.

Le développement de cette partie fera l'objet de travaux de recherches futurs et ne sera pas inclus dans ce mémoire de thèse.

## 4.6 Conclusion

Dans ce chapitre, nous avons présenté en détails une architecture AAA distribuée pour les MANETs que nous avons appelée *Dist-AAA*. Elle a fait l'objet de deux publications [82] [83]. Son initialisation repose, en partie, sur un tiers de confiance tel qu'un opérateur, tandis qu'en matière d'authentification et d'autorisation, elle opère de façon autonome. Le protocole d'authentification qu'elle met en œuvre est extensible grâce à l'emploi d'EAP et distribué grâce à l'application de la cryptographie à seuil. Il permet aussi une authentification mutuelle, nécessaire dans un environnement ouvert comme celui des MANETs, entre un nœud client arrivant et les pairs AAA au sein de Dist-AAA. Sa solidité par rapport à différentes attaques a été expliquée. La validité a été présentée de façon informelle puis un résultat de validité relative conditionnellement à celle de la cryptographie à seuil a été établie au moyen d'une méthode formelle.

Les mécanismes d'autorisation définis sont basés sur l'emploi d'un jeton d'accès en conjonction avec la technique CGA et le protocole sécurisé de découverte de voisins SEND. D'ailleurs, dans le cas d'autorisation aux services de contenus, nous avons étendu les opérations de ce protocole au delà du voisinage immédiat d'un nœud consommateur afin de permettre aux nœuds distributeurs de vérifier la validité de son jeton.

Le protocole d'authentification présenté apparaît comme raisonnablement sûr. Il reste à en établir l'utilité en examinant les temps d'authentification attendus puis les probabilités de réponse aux demandes d'authentification. Nous ferons, pour cela, une étude par modélisation associée à une étude par simulations. Des aspects relevant des principes de simulation, de la qualité du simula-

teur choisi, de la charge du réseau produite par le protocoles d'authentification seront alors abordés. Cela fera l'objet des deux prochains chapitres.

# Chapitre 5

# Validation formelle de la sécurité du protocole AAA d'authentification

## Sommaire


**5.1 Introduction . . . . . 105**
**5.2 Vérification formelle avec AVISPA . . . . . 106**
5.2.1 Modélisation en HLPSL . . . . . 106
5.2.2 Modèle de Dolev-Yao . . . . . 107
5.2.3 L'outil SPAN . . . . . 107
5.2.4 Preuves de sécurité . . . . . 107
**5.3 Spécification du protocole d'authentification . . . 108**
5.3.1 Protocole ISO 9798-3 . . . . . 108
5.3.2 Notre protocole d'authentification . . . . . 108
5.3.2.1 Problèmes rencontrés . . . . . 108
5.3.2.2 Solutions apportées . . . . . 109
5.3.2.3 Modélisation pour $th = n$ . . . . . 110
5.3.2.4 Modélisation pour $th < n$ . . . . . 111
**5.4 Exploitation des résultats . . . . . 112**
5.4.1 Protocole ISO 9798-3 . . . . . 112
5.4.2 Notre protocole . . . . . 114
5.4.2.1 Trois pairs corrompus . . . . . 116
5.4.2.2 Un seul pair corrompu . . . . . 118
**5.5 Conclusion . . . . . 118**


## 5.1 Introduction

Dès lors qu'il s'agit de prouver qu'un programme donné s'exécute correctement, c.-à-d. en gros qu'il fait les choses qu'il doit faire et ne fait pas les choses qu'il ne doit pas faire, les méthodes formelles sont un outil de choix. On peut

avoir une idée de leurs capacités en consultant la page consacrée au logiciel astrée [86] ou un livre de référence consacré à ces méthodes [87].

Lié à ce courant de pensée de la sûreté du logiciel, le domaine de la sûreté des protocoles reste pourtant en décalage. Plusieurs raisons peuvent être avancées pour expliquer ce décalage. La première est que les travaux sur la validation formelle des protocoles ont commencé plus tard. La deuxième est qu'alors qu'un analyseur formel peut prendre directement en charge un programme (en C, en Java, etc.), un protocole ne peut pas constituer l'ensemble des données d'un autre programme. Il a donc fallu décrire les protocoles au moyen de langages de spécification formelle (LSF), appelés en anglais *Formal description techniques* (FDT). L'ISO a mis en place un groupe de travail dans les années 1980 qui a abouti à la standardisation de deux FDT : LOTOS et ESTELLE [88].

En ce qui concerne la sécurité, le décalage est plus important et il a fallu le lancement en 2001 du cinquième programme cadre de la commission européenne [89] pour initier le projet AVISPA (*Automated Validation of Internet Security Protocols and Applications*) [90]. Les principaux résultats étaient publiés dès avant 2006. Alors que les travaux sur Scyther de Cas Cremers [91] présentent une extension de l'algorithme Athena dans un logiciel plus récent, on peut encore considérer AVISPA comme *state-of-the-art*. C'est la raison pour laquelle nous avons choisi cet outil pour étudier notre protocole d'authentification.

## 5.2 Vérification formelle avec AVISPA

Avant de faire le moindre travail de vérification, il y a lieu de donner une spécification formelle du protocole. On va en particulier décrire sous une forme très précise les différents échanges entre les participants. Cette description, quasiment mathématique, se fait au moyen du langage HLPSL (*High Level Protocol Specification Language*). Ce langage permet tout à la fois de modéliser le protocole et de définir celles des propriétés du protocole que l'on souhaite vérifier. On pourra ainsi vérifier la confidentialité d'une donnée ou encore l'authentification de certains participants. C'est sur la base du texte de spécifications en HLPSL que le travail de vérification sera conduit.

### 5.2.1 Modélisation en HLPSL

Le langage HLPSL [92] permet de préciser les différents échanges intervenant dans le protocole en utilisant des variables de différents types. Cependant, ces variables gardent un caractère purement formel : on peut ainsi définir une variable « clé publique » mais il est impossible de lui donner une valeur et même seulement de définir sa taille. Par ailleurs, le langage ne peut pas décrire d'opérations cryptographiques complexes. Les algorithmes tels que ceux de Shamir ou de Shoup, c.-à-d. ceux qui correspondent au partage de clé sont ainsi impossibles à formaliser dans ce langage. Finalement, AVISPA considère les opérations cryptographiques comme une sorte de « boîte noire ». Le chiffrement est parfait et on suppose que les clés ont été suffisamment bien choisies pour que tout fonctionne parfaitement. D'autre part, le comportement de l'attaquant suit également plusieurs règles qui permettent de le modéliser et que nous présentons ci-dessous.

### 5.2.2 Modèle de Dolev-Yao

La formalisation des capacités de l'attaquant est définie dans le modèle de Dolev-Yao [93]. Les deux hypothèses centrales de ce modèle sont : le chiffrement est *parfait* et l'attaquant *est* le réseau. Dire que le chiffrement est parfait revient à dire que l'attaquant ne fait pas de cryptanalyse et donc que les clés sont incassables. Cette hypothèse paraît raisonnable en pratique, à condition que l'implémentation prenne des clés suffisamment longues. D'autre part, on suppose que l'attaquant est incapable de décrypter un message chiffré s'il ne connaît pas la clé de déchiffrement (cf. section 2.4.1 du chapitre 2).

La seconde hypothèse décrit un attaquant omnipotent. Comme cet attaquant est le réseau, il peut, à son gré, bloquer ou intercepter quelque message que ce soit. Il peut également générer des messages à partir des informations qu'il a collectées sur le réseau. Il peut aussi participer simultanément à plusieurs sessions du protocole. Vis à vis du chiffrement, l'attaquant est capable de chiffrer et de déchiffrer dès lors qu'il a connaissance des clés (symétriques, publiques/privées) nécessaires.

### 5.2.3 L'outil SPAN

Une fois que le protocole a été spécifié en HLPSL, on aimerait savoir si la spécification correspond bien à ce qui est attendu. En effet, le langage HLPSL a une syntaxe très permissive ce qui peut autoriser des approximations de codage. En fait, le passage de la description initiale du protocole à la description formalisée, réalisée par un être humain, donc faillible, est un point faible du processus de vérification qu'il y a donc lieu de consolider. On utilise pour cela l'animateur de protocoles de sécurité SPAN (*Security Protocol ANimator*). Celui-ci permet de présenter et d'étudier le comportement des différents agents du protocole à l'aide de diagrammes.

On peut aussi initier des échanges à la main. SPAN calcule alors toutes les transitions possibles. Il est aussi possible de suivre l'évolution des valeurs des variables au cours des transitions. Cela permet de s'assurer que la spécification est bien celle attendue. Lorsque l'on est assuré que la spécification est conforme, on va pouvoir faire des attaques « à la main » à l'aide de SPAN. On peut ainsi jouer le rôle de l'attaquant et tester la robustesse du protocole.

### 5.2.4 Preuves de sécurité

AVISPA permet de découvrir des attaques. Lorsque les propriétés de sécurité du protocole ne sont pas respectées, AVISPA produit systématiquement un exemple d'attaque. On n'obtient donc pas une liste exhaustive des attaques possibles sur le protocole, mais l'on saura qu'une attaque est possible et donc que le protocole ne remplit pas ses objectifs de sécurité.

Par ailleurs, l'utilisation de SPAN permet de tester le fonctionnement du protocole en simulant des cas non prévus par le protocole. Il est ainsi possible de détecter des situations autorisées par le protocole mais qui ne correspondent à rien dans la réalité et dont l'existence peut éventuellement à terme poser un problème de sécurité.

## 5.3 Spécification du protocole d'authentification

Nous allons maintenant décrire comment s'est passée la phase de modélisation : quels ont été les problèmes rencontrés, quelles solutions ont été trouvées et quels ont été les choix de modélisation. Les résultats fournis par AVISPA seront présentés et commentés dans la section 5.4.

### 5.3.1 Protocole ISO 9798-3

Étant donné que le protocole étudié étend le protocole d'authentification décrit dans le standard ISO 9798-3 [80], une première étude, conduite par des étudiants placés sous notre responsabilité [94], a commencé par la modélisation de ce dernier afin de préciser les échanges employés par le protocole étudié. Nous nous référons pour la suite à cette étude, en particulier en ce qui concerne les figures et les notations. Ce plan de travail a permis une prise en main du langage HLPSL sur un exemple simple à spécifier. En effet, la modélisation proposée par l'ISO 9798-3 reste assez élémentaire : une grande partie du travail est déjà accomplie et se trouve disponible sur le site du projet AVISPA [90].

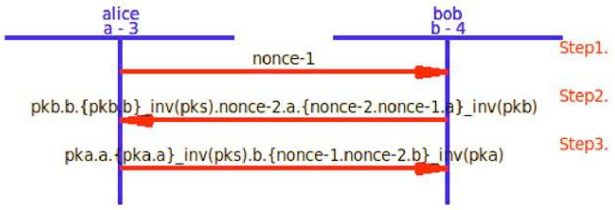


FIGURE 5.1 – Animation SPAN

De plus, ce schéma de protocole ne fait intervenir que deux participants et utilise la cryptographie classique. Il a seulement fallu apporter à ces documents quelques modifications pour que la spécification corresponde à la version de la norme ISO adéquate. Le protocole a ensuite été animé avec SPAN pour voir si la modélisation était correcte (cf. FIGURE 5.1).

### 5.3.2 Notre protocole d'authentification

#### 5.3.2.1 Problèmes rencontrés

On a vu que le protocole étudié reprend en grande partie les mécanismes du protocole ISO 9798-3 (cf. section 4.4.3). Pourtant, sa modélisation a été beaucoup plus complexe que celle de celui-ci. En effet, il a fallu surmonter différents problèmes, majoritairement dus aux limitations de l'outil AVISPA.

Tout d'abord, AVISPA considère la cryptographie comme une boite noire pré-codée dans le langage de description. En conséquence, il est impossible d'effectuer d'autres opérations cryptographiques que celles prévues par le langage. Pour cette raison, les seules opérations prévues sont le chiffrement/déchiffrement par une clé symétrique ou asymétrique ainsi que le hachage (cf. section 2.4 du chapitre 2). De plus, AVISPA ne prend pas en compte les opérations arithmétiques les plus élémentaires comme l'addition. Cette limitation est clairement énoncée dans le manuel et le tutoriel d'AVISPA : « *Do not use arithmetic operators/relations (e.g.'+', '=<'). They are not supported by the translator.* »,

« *HLPSL does not support arbitrary arithmetic operators ; however, we can model an approximation of addition which will reflect the properties that, from a security perspective, are most important* ». Il apparaît donc que les éléments cryptographiques fondamentaux du protocole que sont la combinaison des parts de signature dans le schéma de Shoup [81] ou la vérification d'une adresse CGA, non contents de n'être pas présents dans le langage HLPSL, ne sont même pas programmables dans ce langage. Au mieux, il faut se contenter d'une modélisation approchée de ces processus.

Un autre problème rencontré a été l'impossibilité pour un agent, d'envoyer plusieurs messages en même temps, voire même successivement. Un agent ne peut envoyer qu'un seul message, ce message est reçu par un autre agent et la réception provoque un changement d'état de ce dernier. Cet agent doit attendre la réponse d'un autre agent avant de pouvoir envoyer un nouveau message. On peut faire l'analogie avec des échanges de *ping-pong*.

Il n'a donc pas été possible de coder directement le fait qu'un JN envoie son identifiant à tous les pairs AAA et attende les *th* premières réponses. Dans la modélisation AVISPA, JN ne peut pas envoyer son identifiant à plusieurs pairs en même temps.

#### 5.3.2.2 Solutions apportées

Afin de résoudre le problème de l'envoi simultané de l'identifiant de JN aux pairs, le problème a été modélisé de la façon suivante : JN envoie son identifiant une première fois. Le premier pair, $AAA_1$ reçoit cet identifiant et lui envoie le

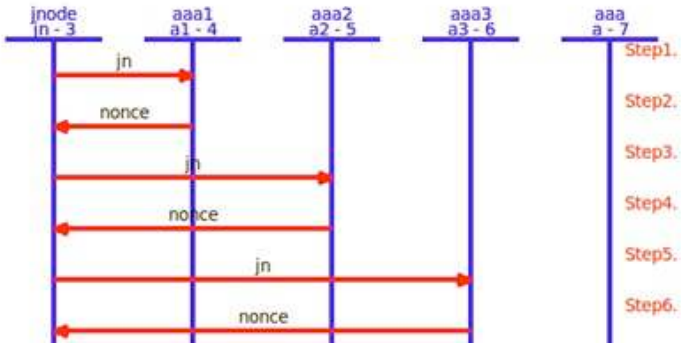


FIGURE 5.2 – Modélisation de l'envoi de $ID_{JN}$ (jn) et de $R_{AAA}$ (*nonce*)

*nonce*[1] $R_{AAA}$. Lorsque JN reçoit $R_{AAA}$, il envoie son identifiant au pair $AAA_2$ et ainsi de suite (cf. FIGURE 5.2). Bien que dans la réalité le protocole ne soit pas implémenté de cette façon, nous considérons que cette modélisation conserve

1. Nonce dans l'acception ecclésiastique se traduit en français par nonce qui est la forme que nous avons retenue. Il s'agit d'un faux-ami. Dans le contexte informatique, ce mot-valise construit sur *number once* pourrait se construire sur nombre unique mais nous n'avons pas pu nous résoudre à le faire et avons gardé une traduction à l'évidence inexacte.

les propriétés de sécurité de ce protocole et reste raisonnablement fidèle à son fonctionnement.

Comme il n'était pas possible de coder la transformation des parts de signature en une signature globale du groupe des pairs AAA, il était nécessaire d'en chercher une modélisation. Il a été choisi d'introduire un agent supplémentaire qui représenterait l'autorité AAA globale. Cet agent est un agent fictif : dans la réalité il n'existe pas.

L'idée est que le JN récupère le jeton signé avec les parts de clés privées des *th* pairs $AAA_i$. Il les relaie ensuite au pair AAA global virtuel qui déchiffre les données avec la clé publique du système des pairs $AAA_i$, puis les chiffre avec sa clé privée. Finalement, le JN reçoit le jeton signé par le pair AAA global.

Cette modélisation peut sembler étrange, mais au final, on a fait simplement appel à l'autorité AAA virtuelle pour effectuer une transformation qui a normalement lieu directement dans le JN. On fait donc passer par le réseau une information complémentaire qui, selon la spécification du protocole, devrait rester dans le JN. Comme l'attaquant a un contrôle total du réseau, on lui donne ainsi des possibilités supplémentaires d'action. Cette modélisation n'écarte donc pas d'attaque sur le protocole.

L'autre problème lié à la cryptographie est la difficulté de modéliser la vérification d'une adresse CGA. Comme les messages échangés à cet effet entre le JN et le destinataire résultent d'échanges SEND, Diffie-Hellman et de données d'authentification (cf. section 4.5), nous avons simplement considéré que l'équivalent d'un échange SEND entre JN et le destinataire avait déjà eu lieu et qu'ainsi le destinataire possédait dans son cache des voisins (cf. section C.3.1 de l'annexe C) la CGA du JN ainsi que la clé publique associée.

Alors que la première modélisation aboutit à renforcer les possibilités d'attaques, la seconde modélisation les diminue. Il n'est donc pas théoriquement sûr que la modélisation globale du protocole ne soit pas plus solide que le protocole initial lui même. Pour autant, ce que l'on sait du protocole SEND [70], sans constituer une véritable preuve est de nature à nous rassurer sur ce point.

#### 5.3.2.3 Modélisation pour $th = n$

Il s'agit de comprendre le protocole d'authentification, c'est à dire d'étudier les échanges entre le JN et les pairs AAA (cf. sectionsect :ptmzt du chapitre 4). Cette étude s'arrête à l'obtention du jeton et ne prend pas en compte la phase d'autorisation où le JN communique avec un destinataire dans le réseau MANET.

Elle teste :

- l'authentification mutuelle entre un JN et les différents pairs sollicités,
- l'authentification du pair AAA virtuel vis-à-vis du JN.

Pour $th = n = 3$, l'animation SPAN de cette modélisation est présentée dans la figure 5.3. Lors des *steps* 1 à 6, JN envoie son identifiant aux pairs $AAA_1$, $AAA_2$ et $AAA_3$. Ces derniers lui répondent tous le même $R_{AAA}$ qui a la valeur *nonce* sur le schéma. Lors des *steps* 7 à 12, JN envoie son certificat assorti de différentes données aux $AAA_i$ afin de se faire authentifier. Une fois cette authentification acquise auprès des pairs $AAA_i$, ceux-ci lui envoient leurs données d'authentification ainsi que les différentes parts du jeton signés avec leurs parts de clé privée respectives.

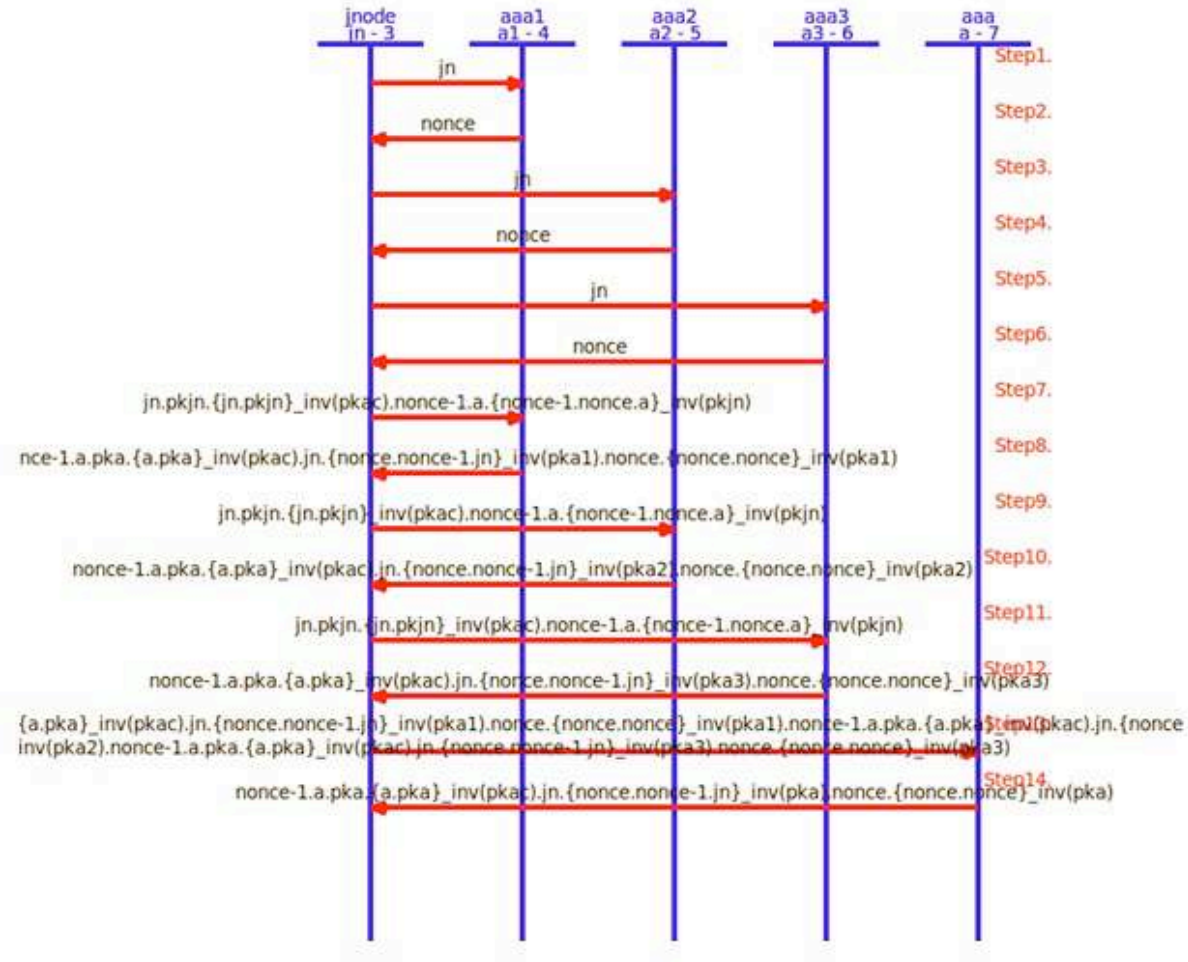


FIGURE 5.3 – Modélisation du protocole pour $th = n = 3$

Comme il a été expliqué plus haut, JN relaie ces messages vers le pair AAA global virtuel pour qu'il combine ces éléments de signatures. Au cours de cette phase, JN ne prend pas connaissance du contenu de ces messages provenant des $AAA_i$ car il ne dispose pas de leurs clés publiques.

Le pair AAA global reçoit les différents messages, combine les signatures et envoie le résultat au JN. JN peut en vérifier la signature grâce à la clé publique du service AAA. Le pair AAA virtuel est donc authentifié auprès du JN.

#### 5.3.2.4 Modélisation pour $th < n$

Par rapport à l'étude pour $th = n$ présentée ci-dessus, l'intérêt de cette modélisation serait la vérification du bon fonctionnement de l'authentification alors même que $th-1$ pairs sont corrompus. Cela pose déjà la question de savoir ce que signifie pour un pair d'être corrompu : s'agit-il d'une simple absence de réponses aux sollicitation ? s'agit-il d'une attitude plus activement néfaste ? Si oui, laquelle ? Quoiqu'il en soit, le langage HLPSL ne semble pas fournir de primitives de description de ces comportements, pas plus qu'il ne fournit les

éléments permettant d'en faire le codage soi-même.

D'autre part, il s'agit d'une certaine façon de tester si le protocole est résistant aux pannes, plus précisément si les pannes de certains éléments du réseau ne sont pas de nature à permettre des attaques contre le protocole. Il semble que le logiciel AVISPA n'est pas été prévu pour traiter de tels problèmes.

## 5.4 Exploitation des résultats

Dans cette section, il s'agit d'exposer et de commenter les résultats que donne AVISPA lorsqu'on lui présente les fichiers décrivant en HLPSL les modélisations précédentes.

### 5.4.1 Protocole ISO 9798-3

Dans un premier temps, AVISPA a été lancé sur la modélisation de ce protocole avec des connaissances assez classiques pour l'attaquant ou intrus (en anglais *intruder*) :

```
intruder_knowledge={a,b,pki,inv(pki),pks,({pki.i}_inv(pks)),
                    ({pka.a}_inv(pks))}
```

L'attaquant a connaissance :
- des agents légitimes (a et b) qui essayent de s'authentifier mutuellement,
- de sa paire de clés publique et privée, `pki` et `inv(pki)`,
- de la clé publique de l'autorité de certification qui a délivré toutes les clés à toutes les entités du réseau, `pks`,
- de son certificat, `({pki.i}_inv(pks))`
- et du certificat d'Alice, `({pka.a}_inv(pks))`.

```
$ avispa iso9798.hlpsl --ofmc
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /mci/ei0710/covacho/avispa-1.1/testsuite/results/iso9798.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.09s
  visitedNodes: 98 nodes
  depth: 8 plies
```

FIGURE 5.4 – Résultat donné par AVISPA

La figure 5.4 montre une impression d'écran du résultat d'AVISPA. Comme on pouvait l'espérer, AVISPA considère que le protocole est « SAFE ».

Un autre test permet, tout en vérifiant l'adéquation de la modélisation à la réalité, de tester le protocole dans des conditions plus dures :

```
intruder_knowledge={a,b,pki,inv(pki),pks,({pki.i}_inv(pks)),
                    ({pka.a}_inv(pks),inv(pka)}
```

On considère maintenant que l'intrus a réussi à se procurer la clé privée de l'agent A, `inv(pka)`. Là encore, le résultat obtenu avec AVISPA est cohérent avec la réalité : l'outil déclare le protocole comme « UNSAFE » car l'attaquant peut s'authentifier auprès de B à la place de A. En effet, comme le montre

```
$ avispa iso9798.hlpsl --ofmc
% OFMC
% Version of 2006/02/13
SUMMARY
  UNSAFE
DETAILS
  ATTACK_FOUND
PROTOCOL
  /mci/ei0710/covacho/avispa-1.1/testsuite/results/iso9798.if
GOAL
  authentication_on_ra
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.01s
  visitedNodes: 2 nodes
  depth: 1 plies
ATTACK TRACE
i -> (b,3): x230
(b,3) -> i: pkb.b.{pkb.b}_inv(pks).Rb(1).a.{Rb(1).x230.a}_inv(pkb)
i -> (b,3): pka.a.{pka.a}_inv(pks).b.{x230.Rb(1).b}_inv(pka)
```

FIGURE 5.5 – Détection de l'attaque

la description de l'attaque figure 5.5, l'attaquant peut signer les messages à la place de A. On peut aussi y voir qu'AVISPA a détecté que la propriété `authentication_on_ra` n'est pas vérifiée.

Le déroulement de l'attaque, illustré par la figure 5.6, est le suivant :

**Step1** L'attaquant envoie un nombre aléatoire `var-x-1` à Bob.

**Step2** Bob envoie à Alice son certificat, un nombre aléatoire `nonce-1`, l'identifiant d'Alice et les deux nombre aléatoires chiffrés avec sa clé privée. Mais le message est intercepté par l'attaquant.

**Step3** L'attaquant renvoie le certificat d'Alice, l'identifiant de Bob et les nombres aléatoires concaténés avec l'identifiant de Bob, chiffrés à l'aide de la clé privée d'Alice.

Comme l'attaquant a réussi à chiffrer le nombre aléatoire `var-x-1` avec la clé privée d'Alice, Bob considère qu'il s'agit bien d'Alice.

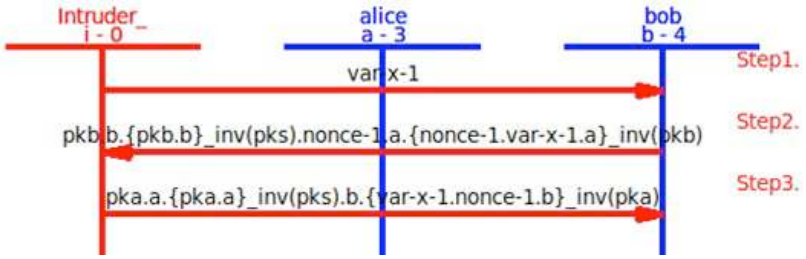


FIGURE 5.6 – Attaque menée par l'intrus

### 5.4.2 Notre protocole

AVISPA a ensuite été lancé sur la modélisation de la section 5.3.2.3 en ne donnant aucune connaissance supplémentaire particulière à l'attaquant.

```
intruder_knowledge={jn, a1, a2, a3,a,inv(pki),pkjn, pka, pka1,
                    pka2, pka3, pkac, pki}
```

Ainsi, l'attaquant a connaissance :
- des différents agents (a, a1, a2, a3),
- de sa paire de clés, `pki` et `inv(pki)`
- et des clés publiques : `pka`, `pka1`, `pka2`, `pka3` et `pkac`.

Contrairement à ce qui était attendu, AVISPA a renvoyé « UNSAFE » pour cette modélisation. Il considère que l'attaquant peut s'authentifier à la place de JN au près de $AAA_2$.

On trouvera figure 5.7 le résultat d'AVISPA et figure 5.8 l'animation qu'en propose SPAN.

Le déroulement est présenté ci-dessous, on a eu soin tout en maintenant la numérotation des « steps » successives présentées par SPAN, de rappeler dans quelle étape du protocole (cf. section 4.4.3 du chapitre 4) elle se situe :

- Étape 1 :
  - Step1, Step5, Step7 - JN envoie son identifiant en direction de $AAA_2$ (en minuscule sur la figure 5.8). L'attaquant, qui se trouve sur le parcours, bloque ce message et le modifie. Il bloque ensuite les messages destinés aux pairs $AAA_1$ et $AAA_3$.
  - Step2 - L'attaquant envoie le message dans lequel il a changé l'adresse IP de l'émetteur en son adresse au pair $AAA_2$.
- Étape 2 :
  - Step3 - Le pair $AAA_2$ envoie un défi (nombre aléatoire) `dummy_nonce` en direction du JN. Ce message parvient en fait à l'attaquant.
  - Step4, Step6, Step8 - L'attaquant envoie à JN les nombres aléatoires en se faisant passer successivement pour $AAA_1$, $AAA_3$ et $AAA_2$.
- Étape 3 :
  - Step9 - JN envoie en direction de chacun des pairs ses informations d'authentification en direction de $AAA_2$. Ce message est intercepté par l'attaquant.

```
$ avispa AAA-v1.6.hlpsl --ofmc
% OFMC
% Version of 2006/02/13
SUMMARY
  UNSAFE
DETAILS
  ATTACK_FOUND
PROTOCOL
  /mci/ei0710/covacho/avispa-1.1/testsuite/results/AAA-v1.6.if
GOAL
  authentication_on_jn_aaa2_rjn
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.05s
  visitedNodes: 16 nodes
  depth: 5 plies
ATTACK TRACE
i -> (jn,3): start
(jn,3) -> i: jn
i -> (a2,3): jn
(a2,3) -> i: dummy_nonce
i -> (jn,3): x268
(jn,3) -> i: jn
i -> (jn,3): x285
(jn,3) -> i: jn
i -> (jn,3): dummy_nonce
(jn,3) -> i: jn.pkjn.{jn.pkjn}_inv(pkac).Rjn(5).a.{Rjn(5).dummy_nonce.a}_inv(pkjn)
i -> (a2,3): jn.pkjn.{jn.pkjn}_inv(pkac).Rjn(5).a.{Rjn(5).dummy_nonce.a}_inv(pkjn)
(a2,3) -> i: Rjn(5).a.pka.{a.pka}_inv(pkac).jn.{dummy_nonce.Rjn(5).jn}_inv(pka2).dummy_nonce.{dummy_nonce.dummy_nonce}_in
```

FIGURE 5.7 – Résultat donné par AVISPA

  - Step10 - L'attaquant envoie ce message en direction de $AAA_2$ en ayant modifié l'adresse de l'émetteur.
- Étape 4 :
  - Step11 - $AAA_2$ croit authentifier JN alors que la source du message est l'attaquant. Il construit un(e part de) jeton correctement adressé(e) à ce dernier et lié(e) cryptographiquement à son adresse.

L'analyse de la faille trouvée appelle plusieurs considérations. Tout d'abord, AVISPA ne prenant pas en compte la multiplicité des pairs, travaille comme s'il s'agit de mettre en défaut un seul d'entre eux (ici $AAA_2$). Il est ainsi capable de mettre en évidence une sorte d'« attaque de l'homme du milieu ». Telle qu'elle est décrite, elle ne pourrait fonctionner dans le protocole présenté à moins que l'attaquant ne puisse se faire passer comme homme du milieu sur chacun des chemins menant du JN à l'un quelconque des pairs. On pourrait dire qu'AVISPA nous a permis de mettre en évidence une attaque possible contre le protocole que nous appellerons l'*attaque de l'homme du centre*. En pratique, celle-ci n'est donc possible que si l'attaquant se trouve placé pendant assez longtemps à un point d'articulation du graphe des routes (cf. section 4.5.1.1 du chapitre 4). Nous considérons, notamment en raison du mouvement, que cette éventualité est hautement improbable.

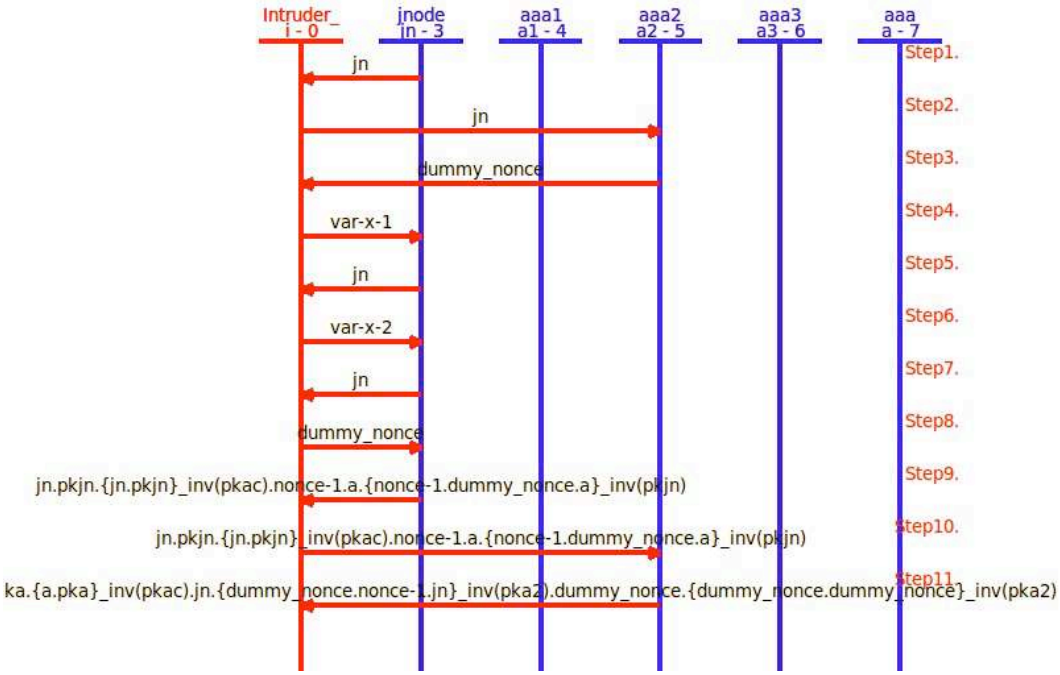


FIGURE 5.8 – Animation SPAN

Pour autant, la version du protocole testée ci-dessus n'en est pas moins falsifiable. L'exigence raisonnable d'une sécurité totale amène à modifier le message transmis aux pairs à l'étape 3. Initialement, il était (cf. section 4.4.3 du chapitre 4) :

$\mathrm{cert_{JN}}$, $\mathrm{R_{JN}}$, R, $\mathrm{ID_i}$, $\mathrm{t_{JN}}$, $\mathrm{sign_{JN}}(\mathrm{R_{JN}}, \mathrm{R_i}, \mathrm{ID_i})$

Nous le modifions en :

$\mathrm{cert_{JN}}$, $\mathrm{R_{JN}}$, R, $\mathrm{ID_i}$, $\mathrm{t_{JN}}$, $\mathrm{sign_{JN}}(\mathrm{R_{JN}}, \mathrm{R_i}, \mathrm{ID_i}, \mathbf{CGA_{JN}})$

Ce simple changement de signature interdit désormais à l'homme du centre de modifier les adresses CGA des messages qu'il intercepte puis transmet. Il est à noter que cette légère modification se fait sans rallongement aucun des messages transmis.

#### 5.4.2.1 Trois pairs corrompus

Pour tester la cohérence de la modélisation, nous avons décidé de faire comme si les trois pairs d'authentification avaient été corrompus. Nous avons donc donné à l'attaquant la connaissance des clés privées des pairs, et donc de la clé du pair AAA global :

```
intruder_knowledge={jn, a1, a2, a3, a, i,d,pkjn, pka, pka1, pka2, pka3, pkac, pki,pkd
```

Comme nous pouvions l'attendre AVISPA déclare le protocole « UNSAFE » et soulève le fait que l'ataquant peut s'authentifier sur JN à la place de AAA :

```
$ avispa AAA-v1.9.hlpsl --ofmc
% OFMC
% Version of 2006/02/13
SUMMARY
  UNSAFE
DETAILS
  ATTACK_FOUND
PROTOCOL
  /mci/ei0710/covacho/avispa-1.1/testsuite/results/AAA-v1.9.if
GOAL
  authentication_on_aaa_jn_raaa
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.32s
  visitedNodes: 131 nodes
  depth: 9 plies
ATTACK TRACE
i -> (a1,3): jn
(a1,3) -> i: dummy_nonce
i -> (jn,3): start
(jn,3) -> i: jn
i -> (jn,3): x286
(jn,3) -> i: jn
i -> (jn,3): x312
(jn,3) -> i: jn
i -> (jn,3): dummy_nonce
(jn,3) -> i: jn.pkjn.{jn.pkjn}_inv(pkac).Rjn(5).a.{Rjn(5).dummy_nonce.a}_inv(pkj
n)
i -> (a1,3): jn.pkjn.{jn.pkjn}_inv(pkac).Rjn(5).a.{Rjn(5).dummy_nonce.a}_inv(pkj
n)
(a1,3) -> i: Rjn(5).a.pka.{a.pka}_inv(pkac).jn.{dummy_nonce.Rjn(5).jn}_inv(pka1)
.dummy_nonce.{ipcgajn.dummy_nonce}_inv(pka1)
i -> (jn,3): x380
(jn,3) -> i: jn.pkjn.{jn.pkjn}_inv(pkac).Rjn(5).a.{Rjn(5).dummy_nonce.a}_inv(pkj
n)
i -> (jn,3): x406
(jn,3) -> i: jn.pkjn.{jn.pkjn}_inv(pkac).Rjn(5).a.{Rjn(5).dummy_nonce.a}_inv(pkj
n)
i -> (jn,3): x432
(jn,3) -> i: x380.x406.x432
i -> (jn,3): Rjn(5).a.pka.{a.pka}_inv(pkac).jn.{dummy_nonce.Rjn(5).jn}_inv(pka).
x459.{ipcgajn.x459}_inv(pka)
(jn,3) -> i: ipcgajn.ipcgad.pkjn.Nonce(10).x459.{ipcgajn.x459}_inv(pka).dummy_no
nce.exp(dummy_nonce,Njn(10)).{ipcgajn.ipcgad.pkjn.Nonce(10).x459.{ipcgajn.x459}_
inv(pka).dummy_nonce.exp(dummy_nonce,Njn(10))}_inv(pkjn)
```

FIGURE 5.9 – Résultat donné par AVISPA

**Steps 1 et 2 :** l'attaquant récupère le nombre aléatoire $R_{AAA}$ (avec la valeur *dummy_nonce*) depuis $AAA_1$.

**Steps 3 à 8 :** L'attaquant se fait passer pour les pairs $AAA_i$ et renvoie RAAA à JN.

**Steps 9 et 10 :** JN envoie à $AAA_1$ son certificat, $R_{JN}$, l'identifiant du pair AAA global et les 2 nombres aléatoires concaténés à cet identifiant, chiffrés avec la clé publique de JN. Mais l'attaquant intercepte ce message.

**Steps 11 à 16 :** L'attaquant envoie des faux messages à JN. JN crois que ces messages contiennent les parts de signatures des pairs $AAA_i$.

**Steps 17** JN relaye les fausses parts de signatures à l'attaquant en pensant communiquer avec le pair AAA.

**Steps 18** L'attaquant renvoie le bon message signé avec la clé privée du pair AAA, et s'authentifie par la même en tant que AAA sur JN.

On voit que les steps 17 et 18 n'existeraient pas dans la réalité. Cependant, le résultat aurait été le même, JN aurait recombiné les signatures de Shoup en *step* 16 et l'attaquant aurait réussi à s'authentifier en tant que AAA.

#### 5.4.2.2 Un seul pair corrompu

Nous avons fait un dernier test en considérant que seul le pair $AAA_1$ avait été corrompu, et en ne donnant donc que la clé privée de $AAA_1$ à l'attaquant :

```
intruder\_knowledge=\{jn, a1, a2, a3, a, i,d,pkjn, pka, pka1, pka2, pka3, pkac, pki,p
```

```
$ avispa AAA-v1.9.hlpsl --ofmc
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /mci/ei0710/covacho/avispa-1.1/testsuite/results/AAA-v1.9.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.24s
  visitedNodes: 112 nodes
  depth: 11 plies
```

FIGURE 5.10 – Résultat donné par AVISPA

On voit sur la figure 5.10 qu'AVISPA renvoie la valeur « SAFE », ce qui correspond bien à ce que l'on attendait de ce protocole distribué.

## 5.5 Conclusion

Tout au long de ce chapitre, nous avons pu mettre en œuvre un outil *state-of-the-art* d'analyse de la sécurité des protocoles. Nous avons ainsi mis en évidence que cet outil n'est pas adapté à l'étude de protocoles à interlocuteurs multiples et à fortiori de protocoles mettant en œuvre une cryptographie à seuil. Cette inadéquation nous a amené à devoir faire des efforts d'ingéniosité pour développer dans le cadre du langage HLPSL une modélisation approchée du protocole étudié. Une telle situation n'est pas souhaitable. Le but des outils de preuve automatique étant par essence d'automatiser le processus.

Malgré ces défauts et les difficultés qui en ont découlé, l'utilisation d'une validation formelle a fait la preuve de sa puissance et par là même de son intérêt

en mettant en évidence dans une version préliminaire du protocole une faiblesse qui nous avait sur l'instant échappé. Nous avons ressenti la réelle nécessité d'un outil formel plus développé mais la recherche menée à travers la littérature ne nous a pas permis de découvrir un tel programme ni même d'articles de recherche théorisant le sujet. Quoiqu'il en soit, au terme de cette partie nous somme désormais en situation de dire que le protocole d'authentification, tel qu'il vient d'être rectifié, est un protocole solide. Il reste à étudier sa mise en œuvre dans un réseau. Cela sera l'objet de la partie suivante.

# Troisième partie

# Évaluation du protocole

# Chapitre 6

# Évaluation du temps d'authentification

## Sommaire


**6.1 Introduction . . . . . 123**
**6.2 Modélisation et simulation : terminologie . . . . . 124**
**6.3 Quelques définitions . . . . . 125**
**6.4 Modèle mathématique et calculs . . . . . 127**
6.4.1 Calcul du temps de transfert sur un lien à un seul saut 128
6.4.2 Calcul du temps de transfert sur un lien à sauts multiples . . . . . 130
6.4.3 Prise en compte du hasard . . . . . 130
6.4.4 Calcul du temps d'authentification . . . . . 133
6.4.5 Résultats . . . . . 135
**6.5 Modèles emboîtés et simulations . . . . . 138**
6.5.1 Considérations techniques, paramétrages et conditions de validation . . . . . 138
6.5.2 Scénario statique . . . . . 139
6.5.3 Ajout de mouvements déterministes . . . . . 143
6.5.4 Prise en compte des temps de calculs cryptographiques 147
**6.6 Conclusion . . . . . 147**


## 6.1 Introduction

Un des indicateurs de performance d'un protocole destiné à fonctionner dans un réseau est la durée de son exécution. Cette valeur est déterminante dans l'adoption de ce protocole ou son abandon. Au chapitre 4, nous avons conçu un protocole d'authentification. Nous avons aussi fait appel au protocole SEND et à l'extension Ext-SEND, que nous avons définie, pour réaliser le service d'autorisation. Dans ce chapitre, nous proposons d'évaluer le temps nécessaire pour accomplir une authentification et nous laissons de côté l'évaluation du temps d'autorisation. La raison de ce choix est liée au mode de communication one-to-many du protocole d'authentification qui est manifestement plus complexe à

appréhender que celui one-to-one utilisé par les protocoles SEND et Ext-SEND. De plus, alors que le protocole d'authentification est à quatre étapes, chaque étape comportant plusieurs messages envoyés ou reçus par les pairs AAA impliqué [1], SEND et Ext-SEND sont à deux étapes seulement, chaque étape consistant en un seul message envoyé ou reçu. Par ailleurs, plusieurs travaux ont déjà porté sur le protocole SEND [70] [95] [96] [97] et, plus généralement, sur des protocoles one-to-one à deux messages, aller et retour, dans les MANETs [98].

S'agissant d'étude de performances impliquant des objets réels devant effectivement fonctionner, il est clair qu'il y aura finalement lieu de les réaliser. Pourtant, l'évolution scientifique et technique de ces cinquante dernières années a toujours consisté à pousser plus loin les calculs, en les prolongeant par des simulations numériques de plus en plus intensives basées sur des modèles de plus en plus réalistes. D'une certaine façon, le temps des maquettes est révolu et les premières réalisations sont extrêmement proches des objets commercialisés. Nous nous inscrivons dans cette même optique et proposons tout d'abord un modèle dont les propriétés s'évaluent simplement au moyen de l'analyse mathématique puis un modèle beaucoup plus proche de la réalité mais dont les propriétés ne sont accessibles qu'à travers la simulation.

## 6.2 Modélisation et simulation : terminologie

Sous des hypothèses suffisamment simplificatrices, on peut écrire une description du système sous forme d'équations, qu'il s'agisse d'équations différentielles, d'équations stochastiques, voire même d'équations différentielles stochastiques. Dans certains cas, cet ensemble d'équations peut être résolu explicitement, soit de façon exacte soit de façon approchée. L'ensemble des équations est souvent appelé un *modèle* du système et les résultats des calculs une *modélisation* de son comportement.

Cependant, dans beaucoup de situations et spécialement dans celles où intervient le hasard une résolution explicite, exacte ou approchée s'avère impossible. La modélisation que l'on a construite peut alors être utilisée pour établir des propriétés générales du système, pour fixer quelques ordres de grandeurs. Elle ne suffit pas pour établir avec précision le comportement de ce système. La meilleure des solutions est alors de réaliser un modèle informatique calquant le modèle précédent et d'en observer le fonctionnement. On parle alors de *simulations*.

Dans bien des cas, le système étudié est multi-échelles, c'est-à-dire qu'il se compose d'un nombre, éventuellement assez grand, de sous-systèmes particuliers, chacun pouvant en contenir d'autres et chacun d'entre eux devant être modélisé. Dans ce cas là, un calcul explicite devient tout simplement impensable et le recours à la simulation une nécessité irrévocable.

Dans le cas qui nous occupe, où l'on étudie un protocole, il y a lieu de le replacer dans le contexte des protocoles environnants, qu'il s'agisse de couches de plus haut ou de plus bas niveau. Il s'agit aussi de vérifier le comportement des ondes électromagnétiques utilisées ce qui suppose de surcroît une modélisation de la propagation de ces ondes. Il s'agit encore de comprendre les arbitrages réalisés par la couche liaison de données, les méthodes de routage : on est typiquement dans le cas d'emboîtement de modèles. Sauf cas très particulier, il

1. Le nombre de pairs AAA impliqués dépend du seuil cryptographique choisi.

est impossible de résoudre le problème de l'étude du comportement d'un tel système par un calcul direct. On a alors recours à un programme de simulation et compte tenu de la multiplicité des sous-systèmes en jeu, nous verrons que le mieux est d'utiliser un simulateur dans lequel ces sous-systèmes ont été préalablement programmés restreignant ainsi le travail à l'écriture d'une nouvelle portion du simulateur prenant en charge le protocole à étudier et à une paramétrisation soigneuse des autres modules du simulateur.

Nous venons de faire la part entre une modélisation mathématique simple débouchant sur des calculs directs, on parlera simplement de *modélisation* et des modélisations plus fines mettant en œuvre des modèles imbriqués dont on ne peut étudier les propriétés que par la seule simulation, on parle alors de *simulations* même s'il est clair que de nombreuses modélisations sous-jacentes sont mises en œuvre.

## 6.3 Quelques définitions

La version optimisée du protocole d'authentification met en œuvre un protocole en quatre étapes (cf. section 4.4.4 du chapitre 4). Les requêtes envoyées à la première étape ne sont pas forcément reçues au même moment par les pairs AAA. Pour cette raison, des réponses envoyées à la deuxième étape peuvent être émises alors que certaines requêtes ne sont pas encore arrivées à destination. Ce phénomène peut aussi avoir lieu entre la troisième et la quatrième étapes. En revanche, il ne se produit pas entre la deuxième et la troisième étapes car un client doit attendre toutes les réponses des pairs c.-à-d. que la deuxième étape soit complètement terminée avant d'entamer la troisième étape.

Pour des raisons à la fois de précision et de simplification de langage tout au long de ce chapitre, nous définissons les termes suivants :

**Définition 6.3.1.** *La « première phase » est l'ensemble de la première et de la deuxième étapes du protocole d'authentification et la « seconde phase » est l'ensemble de sa troisième et de sa quatrième étapes (cf.* FIGURE *6.1).*

Nous définissons aussi les termes suivants qui seront utiles dans la description du modèle :

**Définition 6.3.2.** *Le « temps de traitement » d'un message est l'intervalle de temps nécessaire à sa construction par la source ou à son traitement par la destination.*

**Définition 6.3.3.** *Le « temps de propagation » est l'intervalle de temps entre le moment où un bit est émis par la source sur le lien physique et le moment où il est reçu par la destination.*

**Définition 6.3.4.** *Le « temps de transmission » est l'intervalle de temps entre le moment où un paquet entre dans la couche liaison de données de la source et le moment où il arrive dans la couche liaison de données de la destination.*

**Définition 6.3.5.** *Le « temps de transfert » d'un message est l'intervalle de temps entre le début de sa construction par la source et la fin de son traitement par la destination. On l'appelle aussi le* temps de bout en bout.

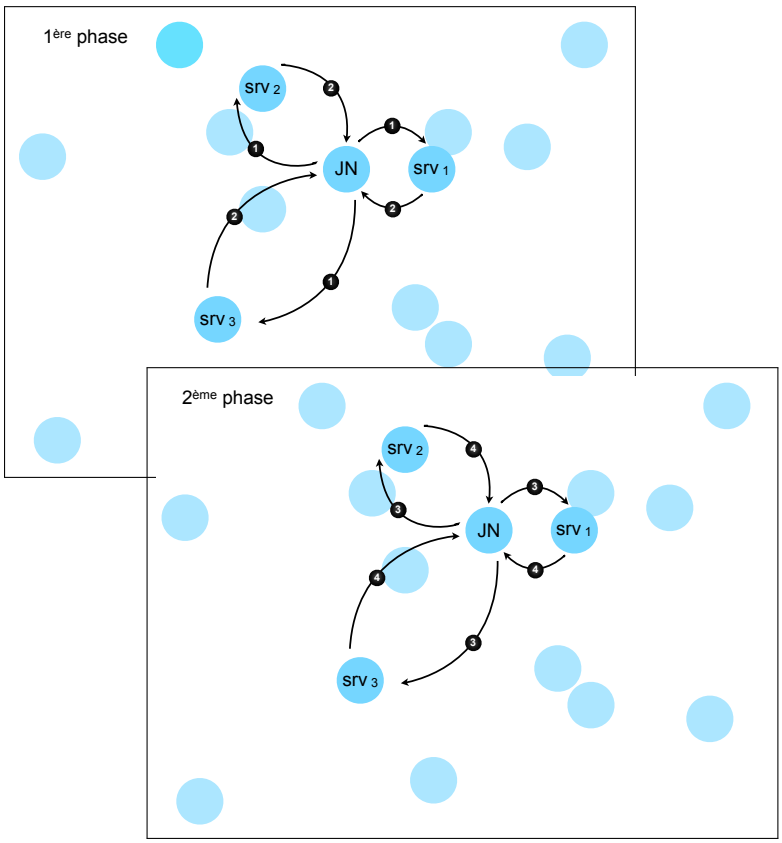


FIGURE 6.1 – Protocole d'authentification à 4 types de messages

**Définition 6.3.6.** *Le « temps d'aller-retour » est le temps de transfert d'une requête émise par un client à destination d'un pair AAA ajouté au temps de transfert de la réponse correspondante envoyée par ce pair à destination du client.*

**Définition 6.3.7.** *Le « temps d'aller-retours global » est la somme des temps d'aller-retour pendant une phase du protocole d'authentification.*

**Définition 6.3.8.** *Le « temps d'authentification » est la durée d'exécution de la version optimisée du protocole d'authentification décrite à la section 4.4.4 du chapitre 4. C'est l'intervalle entre le moment où un client entre dans la première étape et le moment où il finit d'exécuter la quatrième étape. C'est donc la somme de deux temps d'aller-retours globaux.*

**Définition 6.3.9.** *La « longueur d'une route » est définie comme le nombre de sauts qui la composent. On parle aussi de la « longueur d'un lien » séparant deux nœuds. Elle est égale à la longueur de la route qui les rejoint.*

**Définition 6.3.10.** *La « distance entre deux nœuds » est égale à la longueur de la route qui les rejoint.*

## 6.4 Modèle mathématique et calculs

La présente section décrit le raisonnement que nous avons suivi pour construire un modèle rudimentaire permettant de calculer explicitement le temps d'authentification. Afin qu'il soit accessible au calcul tout en restant raisonnablement réaliste, un ensemble d'hypothèses simplificatrices seront émises et discutées tout au long de cette section. Nous en donnerons un résumé à la fin de celle-ci (cf. tableau 6.2).

D'abord, rappelons les notations du chapitre 4 : $n$ est le nombre total de pairs AAA, $th$ le seuil cryptographique, $\mathrm{srv}_1$, $\mathrm{srv}_2$, ..., $\mathrm{srv}_n$ les pairs AAA qui sont aussi serveurs AAA et JN un nœud arrivant client. Nous partons d'un MANET modélisé par un graphe (cf. section 1.2.1 du chapitre 1) dont la géométrie respecte les hypothèses suivantes :

1. Les nœuds ad hoc ne se déplacent pas au cours du temps de vie du réseau. Celui-ci est considéré donc comme statique. Les routes entre chaque pair de nœuds sont par conséquent fixes. Nous les représenterons par des segments de droites à lignes interrompues (cette hypothèse reste approximativement vraie lorsque les mouvements des nœuds sont suffisamment lents pour être d'une amplitude négligeable au cours d'une authentification).

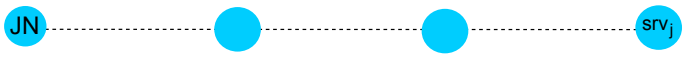


FIGURE 6.2 – Route à 3 sauts, de longueur 3 ; nœuds également espacés

2. Une route peut comporter zéro, un ou plusieurs nœuds relais. Sur les figures, ces nœuds relais seront représentés sur les routes. De plus, puisque le temps de propagation des ondes électromagnétiques entre deux nœuds successifs est insignifiant (3 $\mu s$/km) à cause de la vitesse de ces ondes dans l'air, ce temps peut être considéré comme indépendant de la distance qui les sépare. Les nœuds relais seront donc représentés également espacés le long d'une route qui apparaîtra ainsi coupées en segments plus courts. Ces segments sont en fait ce que l'on appelle des *sauts* (en anglais *hops*) (cf. FIGURE 6.2).
3. Dans un MANET réel, les routes séparant JN de chacun des serveurs $\mathrm{srv}_i$ n'ont pas forcément la même longueur. Pour cela, on prend parmi elles la route la plus longue et on suppose que la longueur de chacune des autres routes est égale à celle-ci. Ceci permettra de calculer une borne supérieure du temps d'authentification, ce qui peut être suffisant dans un premier temps.
4. Ainsi, le modèle simplifié obtenu apparaît comme un graphe étoilé à branches d'égales longueurs (cf. FIGURE 6.3), ce qui n'est pas une représentation très fidèle de la géométrie d'un MANET quelconque, mais suffisant pour mener un raisonnement approché.

En vue de calculer le temps d'authentification d'un nœud arrivant JN, nous calculons d'abord le temps de transfert d'une requête *req* envoyée par JN à un pair AAA $\mathrm{srv}_j$. A cette fin, nous calculons ce temps lorsque JN et $\mathrm{srv}_j$ sont à portée l'un de l'autre puis nous généralisons le calcul à un lien à sauts multiples. Nous déduisons, par la suite, le temps d'aller-retour de *req* et de sa réponse *rep*

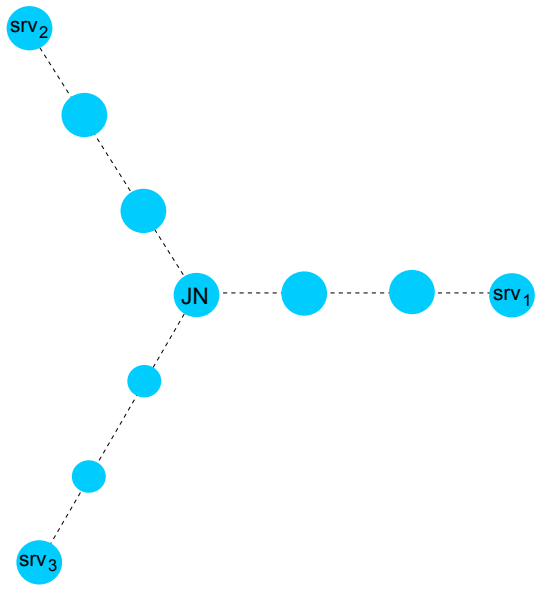


FIGURE 6.3 – Graphe étoilé à branches d'égales longueurs : JN est à la même longueur de route de chaque pair AAA

envoyée par $\mathrm{srv_j}$. Le calcul du temps moyen d'aller-retours global de la première phase est ensuite inféré. Le temps moyen d'aller-retours global de la seconde phase est établi de la même manière. Il suffit de faire la somme de ces deux temps moyens pour trouver, enfin, le temps moyen d'authentification.

### 6.4.1 Calcul du temps de transfert sur un lien à un seul saut

Lorsque le JN et $\mathrm{srv_j}$ sont à portée directe, la distance qui les sépare est égale à un. Pour mener l'étude lorsqu'ils ne sont pas à portée l'un de l'autre, il suffira d'appliquer le raisonnement sur chacun des sauts de la route qui les rejoint.

La figure 6.4 illustre le cheminement de la requête *req* entre le JN et $\mathrm{srv_j}$ :

1. La couche application du JN construit le message EAP de la requête *req*. S'il s'agit d'une requête envoyée à la première étape, cela consiste à y mettre l'identité du JN. Alors que s'il s'agit de la troisième étape, cela consiste à générer un nombre aléatoire et une signature et à les mettre avec d'autres informations dans le message EAP. Le temps de traitement nécessaire à cette construction est noté $t_{\mathrm{rtm_{JN}}}$.
2. Le message EAP obtenu passe successivement par la couche transport et par la couche réseau. Le temps passé dans chacune de ces couches est supposé négligeable. Le paquet obtenu entre, par la suite dans la file d'attente de la couche liaison de données.
3. La trame obtenue est transmise à travers le lien sans-fil. On note le temps de transmission $t_{\mathrm{rsm}}$. Il inclut le temps d'attente dans la file, le temps d'émission de la trame, le temps de propagation du signal, les temps de

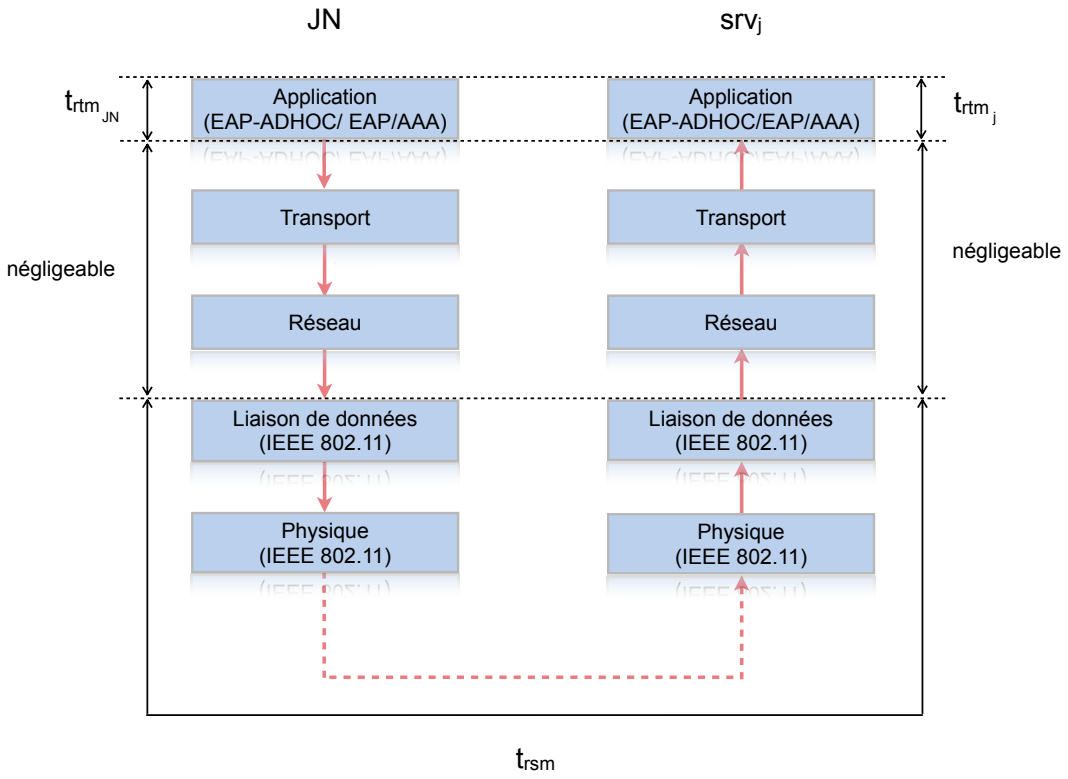


FIGURE 6.4 – Étapes de transmission entre JN et $srv_j$ distants d'un seul saut et temps associés

retransmission en cas de collisions [26] avec d'autres trames transmises sur le même lien et le temps de réception de la trame.

4. Arrivée dans la couche liaison de données de la destination, la trame passe par les couches réseau et transport. Le temps passé dans ces couche est négligeable.
5. Le contenu du message EAP reçu par la couche application est examiné. S'il s'agit de la première étape, $srv_j$ vérifie que l'identité du JN existe bien dans sa base des utilisateurs. S'il s'agit plutôt de la troisième étape, la validité de la signature est vérifiée (cf. section B.5.2 du chapitre B). On note $t_{rtm_j}$ le temps de traitement du message.

Ainsi, le temps de transfert de *req*, noté $t_{rsf}$ sur un lien à un seul saut est :

$$t_{rsf} = t_{rtm_{JN}} + t_{rsm} + t_{rtm_j} \tag{6.1}$$

On suppose que les temps de traitement par la source JN ou par la destination $srv_j$ sont négligeables. En particulier les opérations de calculs nécessaires à

la génération d'une signature par le JN ou à sa vérification par $\mathrm{srv_j}$ sont considérées suffisamment rapides pour se permettre de négliger les temps $\mathrm{t_{rtm_{JN}}}$ et $\mathrm{t_{rtm_j}}$, d'où :

$$t_{rsf} = t_{rsm} \tag{6.2}$$

On suppose, par ailleurs, que, dans les files d'attente des couches de liaison de données, il n'existe pratiquement pas d'autres trames à part celles contenant *req*. Les temps d'attente sont donc négligeables. De plus, étant donnée la vitesse des ondes électromagnétiques dans l'air, le temps de propagation est de l'ordre de 3 $\mu s$/km. Il est donc aussi négligeable. Le temps de transmission se résume donc aux temps d'émission, de retransmission et de réception.

### 6.4.2 Calcul du temps de transfert sur un lien à sauts multiples

Considérons maintenant un lien à sauts multiples séparant JN de $\mathrm{srv_j}$. Soit *hops* la longueur de sa route. La figure 6.5 illustre un exemple pour $hops = 3$.

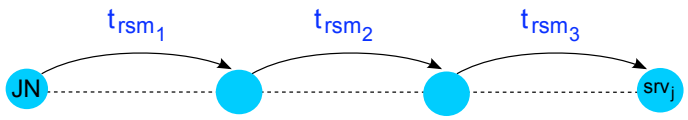


FIGURE 6.5 – Route à 3 sauts, de longueur 3 ; nœuds également espacés

Comme les nœuds sont fixes, les routes le sont aussi. Les opérations de routage ne nécessite pas la mise en place de route et durent un temps que l'on peut supposer donc négligeable. Dans l'exemple, le temps de transfert d'une requête *req* peut alors s'écrire comme ceci :

$$t_{rsf} = t_{rsm_1} + t_{rsm_2} + t_{rsm_3} \tag{6.3}$$

Plus généralement, pour une valeur quelconque du paramètre *hops*, on a :

$$t_{rsf} = \sum_{k=1}^{hops} t_{rsm_k} \tag{6.4}$$

### 6.4.3 Prise en compte du hasard

Le temps $\mathrm{t_{rsm_k}}$ n'est pas déterministe et est soumis aux aléas de transmissions sur le lien sans fil ainsi que nous l'avons présenté à la section 1.4.1 du chapitre 1. $\mathrm{t_{rsm_k}}$ est donc une variable aléatoire positive. On note $\mathrm{F_{t_{rsm,k}}}$ sa fonction de distribution. Elle prend en considération les possibles retransmissions de trames suivant le mécanisme d'accès de base de la fonction DCF (*Distributed Coordination Function*) (cf. annexe D). Le nombre maximum de retransmissions est fixé par la norme IEEE 802.11 [26] à sept.

Soit $p$ la probabilité de retransmission d'une trame et $X$ le nombre de retransmissions. $X$ n'est pas déterministe, c'est donc une variable aléatoire discrète

dont l'univers d'éventualités est $\Omega = \{1, ..., 7\}$. On a alors :

$$\begin{cases} P(X = i) = p^i(1-p) \ pour \ 0 \leqslant i \leqslant 6 \\ P(X = 7) = p^7 \\ P(X = i) = 0 \ pour \ i \geqslant 8 \end{cases} \tag{6.5}$$

D'après le théorème des probabilités totales [99], on a la formule suivante :

$$F_{t_{rsm,k}}(t) = P(t_{rsm_k} \leqslant t) = \sum_{i=0}^{7} P(\{t_{rsm_k} \leqslant t\}|\{X = i\}) \cdot P(\{X = i\}) \tag{6.6}$$

Étant données le schéma de temporisation exponentiel (en anglais *exponential backoff scheme*) présentées à l'annexe D et suivant les formules établies au tableau 6.1, on a :

$$\begin{aligned} P(\{t_{rsm_k} \leqslant t\}|\{X = i\}) &= (i+1)DIFS + (i+1) \cdot P(\{EMd \leqslant t\}) \\ &\quad + \frac{1}{2}\sum_{j=0}^{i-1} 2^j \cdot CW_{min}\theta + i \ ACK_Timeout \\ &\quad + SIFS + EMd_{ACK} \end{aligned} \tag{6.7}$$

où DIFS, $\theta$, SIFS et ACK_Timeout sont les temporisateurs (en anglais *timer*) de la fonction DCF, $\mathrm{CW_{min}}$ est la fenêtre de contention minimale (cf. annexe D), $\mathrm{EMd_{ACK}}$ est le temps nécessaire à l'émission d'un message ACK à un débit égal à 1 Mbps [26] et EMd le temps d'émission d'un message de $l$ octets à un débit égal à $\lambda$.

On suppose que le temps d'émission nécessaire pour transmettre un octet est une variable aléatoire continue positive qui suit une loi exponentielle de paramètre $\lambda$, le débit moyen en octets. Son espérance est donc $1/\lambda$ et sa variance $1/\lambda^2$. Ainsi, le temps moyen nécessaire pour transmettre $l$ octets est $l/\lambda$. Comme $l$ indique la longueur en octets d'un message d'authentification, quelle que soit sa valeur, elle est suffisamment grande pour pouvoir appliquer le théorème de la limite centrée [100]. Ainsi, le temps d'émission de $l$ octets est une variable aléatoire continue positive qui suit une loi gaussienne de moyenne $l/\lambda$ et de variance $l/\lambda^2$ [100].

D'où : $\forall i \in \{0, .., 7\}$ , $t_{rsm_k}$ est, conditionnellement à {X=i}, une variable aléatoire positive qui suit une distribution gaussienne de moyenne $\mu_i$ et de variance $\sigma_i^2$. En utilisant le logiciel de calculs MAPLE (*MAPLE Computer Algebra System* (CAS)) dans toute la suite des opérations (cf. section E.1 de l'annexe E), on obtient :

$$\begin{aligned} \mu_i &= (i+1)DIFS + (i+1)\frac{l}{\lambda} + \frac{1}{2}\sum_{j=0}^{i-1} 2^j \cdot CW_{min}\theta + i \ ACK_Timeout \\ &\quad + SIFS + EMd_{ACK} \end{aligned} \tag{6.8}$$

et

$$\sigma_i^2 = (i+1) \cdot \frac{l}{\lambda^2} \tag{6.9}$$

Tableau 6.1 – Calculs de probabilités compte tenu du schéma de temporisation exponentiel

| Événem$^t$ | Probabilité | Événem$^t$/Cond. | Probabilité conditionnelle |
|---|---|---|---|
| $X = 0$ | $P(X = 0) = 1 - p$ | $\{t_{rsm_k} \leqslant t\}$ $\{X = 0\}$ | $P(\{t_{rsm_k} \leqslant t\}\|\{X = 0\}) = DIFS$ $+ P(\{EMd \leqslant t\}) + SIFS + EMd_{ACK}$ |
| $X = 1$ | $P(X = 1) = p(1 - p)$ | $\{t_{rsm_k} \leqslant t\}$ $\{X = 1\}$ | $P(\{t_{rsm_k} \leqslant t\}\|\{X = 1\}) = DIFS$ $+ 2\ P(\{EMd \leqslant t\}) + \frac{1}{2}CW_{min}\theta$ $+ DIFS + ACK_Timeout + SIFS + EMd_{ACK}$ |
| $X = 2$ | $P(X = 2) = p^2(1 - p)$ | $\{t_{rsm_k} \leqslant t\}$ {X=2} | $P(\{t_{rsm_k} \leqslant t\}\|\{X = 2\}) = 3 \cdot DIFS$ $+ 3\ P(\{EMd \leqslant t\}) + \frac{1}{2}\sum_{j=0}^{1} 2^j \cdot CW_{min}\theta$ $+ 2\ ACK_Timeout + SIFS + EMd_{ACK}$ |
| $X = 3$ | $P(X = 3) = p^3(1 - p)$ | $\{t_{rsm_k}\| \leqslant t\}$ $\{X = 3\}$ | $P(\{t_{rsm_k} \leqslant t\}\|\{X = 3\}) = 4 \cdot DIFS$ $+ 4\ P(\{EMd \leqslant t\}) + \frac{1}{2}\sum_{j=0}^{2} 2^j \cdot CW_{min}\theta$ $+ 3\ ACK_Timeout + SIFS + EMd_{ACK}$ |
| $X = 4$ | $P(X = 4) = p^4(1 - p)$ | $\{t_{rsm_k} \leqslant t\}$ $\{X = 4\}$ | $P(\{t_{rsm_k} \leqslant t\}\|\{X = 4\}) = 5 \cdot DIFS$ $+ 5\ P(\{EMd \leqslant t\}) + \frac{1}{2}\sum_{j=0}^{3} 2^j \cdot CW_{min}\theta$ $+ 4\ ACK_Timeout + SIFS + EMd_{ACK}$ |
| $X = 5$ | $P(X = 4) = p^5(1 - p)$ | $\{t_{rsm_k} \leqslant t\}$ $\{X = 5\}$ | $P(\{t_{rsm_k} \leqslant t\}\|\{X = 5\}) = 6 \cdot DIFS$ $+ 6\ P(\{EMd \leqslant t\}) + \frac{1}{2}\sum_{j=0}^{4} 2^j \cdot CW_{min}\theta$ $+ 5\ ACK_Timeout + SIFS + EMd_{ACK}$ |
| $X = 6$ | $P(X = 6) = p^6(1 - p)$ | $\{t_{rsm_k} \leqslant t\}$ {X=6} | $P(\{t_{rsm_k} \leqslant t\}\|\{X = 6\}) = 7 \cdot DIFS$ $+ 7\ P(\{EMd \leqslant t\}) + \frac{1}{2}\sum_{j=0}^{5} 2^j \cdot CW_{min}\theta$ $+ 6\ ACK_Timeout + SIFS + EMd_{ACK}$ |
| $X = 7$ | $P(X = 7) = p^7(1 - p)$ | $\{t_{rsm_k} \leqslant t\}$ $\{X = 7\}$ | $P(\{t_{rsm_k} \leqslant t\}\|\{X = 7\}) = 8 \cdot DIFS$ $+ 8\ P(\{EMd \leqslant t\}) + \frac{1}{2}\sum_{j=0}^{6} 2^j \cdot CW_{min}\theta$ $+ 7\ ACK_Timeout + SIFS + EMd_{ACK}$ |

Par conséquent, en utilisant la fonction classique erf [101], on obtient :

$$F_{t_{rsm,k}}(t) = \frac{1}{2} - \frac{1}{2}\sum_{i=0}^{6} p^i(1-p)\cdot \mathrm{erf}(\frac{1}{2}\frac{\sqrt{2}(t-\mu_i)}{\sigma_i}) - \frac{1}{2}p^7 \cdot \mathrm{erf}(\frac{1}{2}\frac{\sqrt{2}(t-\mu_7)}{\sigma_7}) \tag{6.10}$$

de moyenne :

$$\mu_{t_{rsm}} = \sum_{i=0}^{6} p^i(1-p)\mu_i + p^7\mu_7 \tag{6.11}$$

et de variance :

$$\begin{aligned}
\sigma^2_{t_{rsm}} \quad = \quad & (-\mu_6^2 + 2\mu_7\mu_6 - \mu_7^2)p^{14} + (-2\mu_7\mu_6 + 2\mu_6^2 - 2\mu_6\mu_5 + 2\mu_7\mu_5)p^{13} \\
& + (-2\mu_6\mu_4 - 2\mu_7\mu_5 + 4\mu_6\mu_5 - \mu_6^2\mu_5^2 + 2\mu_7\mu_4)p^12 + (-2\mu_7\mu_4 \\
& + 2\mu_5^2 - 2\mu_6\mu_3 - 2\mu_6\mu_5 - 2\mu_5\mu_4 + 4\mu_6\mu_4 + 2\mu_7\mu_3)p^11 + (-2\mu_7 \\
& \mu_3 + 2\mu_7\mu_2 - 2\mu_6\mu_2 + 4\mu_6\mu_3 - \mu_5^2 - 2\mu_6\mu_4 + 4\mu_5\mu_4 - \mu_4^2 \\
& - 2\mu_5\mu_3)p^10 + (2\mu_7\mu_1 - 2\mu_4\mu_3 - 2\mu_7\mu_2 + 2\mu_4^2 - 2\mu_6\mu_3 + 4\mu_6 \\
& \mu_2 + 4\mu_5\mu_3 - 2\mu_5\mu_2 - 2\mu_5\mu_4 - 2\mu_6\mu_1)p^9 + (4\mu_4\mu_3 - \mu_3^2 + 4 \\
& \mu_6\mu_1\mu_4^2 + 4\mu_5\mu_2 - 2\mu_6\mu_0 + 2\mu_7\mu_0 - 2\mu_5\mu_3 - 2\mu_5\mu_1 \\
& - 2\mu_4\mu_2 - 2\mu_6\mu_2 - 2\mu_7\mu_1)p^8 + (4\mu_6\mu_0\sigma_6^2 - 2\mu_6\mu_1 - 2\mu_7 \\
& \mu_0 - \mu_6^2 + \mu_7^2 - 2\mu_4\mu_3 + 4\mu_5\mu_1 - 2\mu_5\mu_0 + 4\mu_4\mu_2 - 2\mu_5\mu_2 \\
& - 2\mu_3\mu_2 + \sigma_7^2 + 2\mu_3^2 - 2\mu_4\mu_1)p^7 + (\mu_6^2 - 2\mu_3\mu_1 - \mu_5^2 + 4\mu_5 \\
& \mu_0 + \sigma_6^2 - \mu_3^2 - \mu_2^2 - \sigma_5^2 - 2\mu_6\mu_0 - 2\mu_4\mu_2 + 4\mu_3\mu_2 + 4\mu_4 \\
& \mu_1 - 2\mu_5\mu_1 - 2\mu_4\mu_0)p^6 + (\sigma_5^2 + 4\mu_4\mu_0 - 2\mu_3\mu_0 - 2\mu_5\mu_0 \\
& - 2\mu_2\mu_1 + \mu_5^2 + 4\mu_3\mu_1 - 2\mu_3\mu_2 - 2\mu_4\mu_1 + 2\mu_2^2 - \mu_4^2 - \sigma_4^2)p^5 \\
& + (-2\mu_3\mu_1 - \mu_1^2 - \mu_2^2 - \mu_3^2 + \sigma_4^2 + \mu_4^2 - 2\mu_4\mu_0 + 4\mu_2\mu_1 - \sigma_3^2 \\
& - 2\mu_2\mu_0 + 4\mu_3\mu_0)p^4 + (-2\mu_3\mu_0 + 4\mu_2\mu_0 + \sigma_3^2 - 2\mu_1\mu_0 + 2\mu_1^2 \\
& - 2\mu_2\mu_1\mu_2^2 + \mu_3^2 - \sigma_2^2)p^3 + (-\sigma_1^2 - 2\mu_1^2 + \sigma_2^2 + \mu_2^2 + 4\mu_1 \\
& \mu_0 - 2\mu_2\mu_0 - \mu_0^2)p^2 + (\sigma_1^2 + \mu_1^2 - 2\mu_1\mu_0 + \mu_0^2\sigma_0^2)p + \sigma_0^2
\end{aligned} \tag{6.12}$$

$\mu_{t_{rsm}}$ et $\sigma^2_{t_{rsm}}$ sont indépendantes de $k$ car tous les liens sans-fil sont considérés identiques.

En conséquence, les variables aléatoires positives $\{t_{rsm_k}\}_{1\leqslant k\leqslant hops}$ suivent la même loi de probabilité. Puisque la transmission d'une trame sur un lien d'un saut spécifique est indépendante de la transmission de cette même trame sur un autre lien d'un autre saut, ces variables aléatoires sont indépendantes et le théorème de la limite centrée [100] s'applique à nouveau. D'où, étant donnée l'équation 6.4, $t_{rsf}$ suit une loi gaussienne de moyenne :

$$\mu_{t_{rsf}} = hops \cdot \mu_{t_{rsm}} \tag{6.13}$$

et de variance :

$$\sigma^2_{t_{rsf}} = hops \cdot \sigma^2_{t_{rsm}} \tag{6.14}$$

### 6.4.4 Calcul du temps d'authentification

Soient $t_{rsf,1,j}$ le temps de transfert d'une requête envoyée à la première étape à destination de $\mathrm{srv_j}$, $t_{rsf,2,j}$ le temps de transfert de la réponse correspondante

envoyée pendant la deuxième étape, $t_{rsf,3,j}$ le temps de transfert de la deuxième requête envoyée pendant la troisième étape à destination de srv$_j$ et $t_{rsf,4,j}$ le temps de transfert de la réponse correspondante envoyée à la quatrième étape. Ces temps de transferts ont chacun la même fonction de distribution que $t_{rsf}$ où il suffit de remplacer $l$ par la longueur de la requête ou de la réponse qui convient, soient, dans l'ordre, $l_1$, $l_2$, $l_3$ et $l_4$.
Leurs moyennes respectives sont :

$$\begin{aligned} \mu_{t_{rsf,1,j}} &= hops \cdot \mu_{t_{rsm,1,j}} \ , \\ \mu_{t_{rsf,2,j}} &= hops \cdot \mu_{t_{rsm,2,j}} \ , \\ \mu_{t_{rsf,3,j}} &= hops \cdot \mu_{t_{rsm,3,j}} \ , \\ \text{et } \mu_{t_{rsf,4,j}} &= hops \cdot \mu_{t_{rsm,4,j}} \end{aligned} \tag{6.15}$$

Leurs variances respectives sont :

$$\begin{aligned} \sigma^2_{t_{rsf,1,j}} &= hops \cdot \sigma^2_{t_{rsm,1,j}} \ , \\ \sigma^2_{t_{rsf,2,j}} &= hops \cdot \sigma^2_{t_{rsm,2,j}} \ , \\ \sigma^2_{t_{rsf,3,j}} &= hops \cdot \sigma^2_{t_{rsm,3,j}} \ , \\ \text{et } \sigma^2_{t_{rsf,4,j}} &= hops \cdot \sigma^2_{t_{rsm,4,j}} \end{aligned} \tag{6.16}$$

Soit $t_{ar,1,j} = t_{rsf,1,j} + t_{rsf,2,j}$ le temps d'aller-retour entre JN et srv$_j$ à la première phase et $t_{ar,2,j} = t_{rsf,3,j} + t_{rsf,4,j}$ le temps d'aller-retour à la seconde phase. Les temps $\{t_{ar,1,j}\}_{1 \leqslant j \leqslant n}$ sont différents pour chaque pair AAA car les transmissions entre JN et les pairs, compte tenu du hasard, ne peuvent pas se déroulées toutes de la même façon. Cette remarque est valable aussi pour les temps $\{t_{ar,2,j}\}_{1 \leqslant j \leqslant n}$.

Toutefois, les temps $\{t_{ar,1,j}\}_{1 \leqslant j \leqslant n}$ suivent, tous, la même loi de probabilité. Ils ont donc la même fonction de distribution gaussienne $F_{t_{ar,1}}$ de moyenne :

$$\mu_{t_{ar,1}} = \mu_{t_{rsf,1,j}} + \mu_{t_{rsf,2,j}} \tag{6.17}$$

et de variance :

$$\sigma^2_{t_{ar,1}} = \sigma^2_{t_{rsf,1,j}} + \sigma^2_{t_{rsf,2,j}} \tag{6.18}$$

telle que :

$$F_{t_{ar,1}}(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}\sigma_{t_{ar,1}}} \exp(\frac{-(t-\mu_{t_{ar,1}})^2}{2\sigma^2_{t_{ar,1}}})dt \tag{6.19}$$

Le même raisonnement s'applique aussi aux temps $\{t_{ar,2,j}\}_{1 \leqslant j \leqslant n}$. $F_{t_{ar,2}}$ est de moyenne :

$$\mu_{t_{ar,2}} = \mu_{t_{rsf,3,j}} + \mu_{t_{rsf,4,j}} \tag{6.20}$$

et de variance :

$$\sigma^2_{t_{ar,2}} = \sigma^2_{t_{rsf,3,j}} + \sigma^2_{t_{rsf,4,j}} \tag{6.21}$$

telle que :

$$F_{t_{ar,2}}(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}\sigma_{t_{ar,2}}} \exp(\frac{-(t-\mu_{t_{ar,2}})^2}{2\sigma^2_{t_{ar,2}}})dt \tag{6.22}$$

Soit $t_{arg,1}$ le temps d'aller-retours global nécessaire pour terminer la première phase. Comme JN doit attendre que toutes les réponses des pairs AAA soient arrivées et que les aller-retours des requêtes et des réponses correspondantes s'exécutent en parallèles, $t_{arg,1}$ est le maximum des temps $\{t_{ar,1,j}\}_{1\leqslant j\leqslant n}$ :

$$t_{arg,1} = max\{t_{ar,1,j}\}_{1\leqslant j\leqslant n} \tag{6.23}$$

De ce fait, nous appelons le modèle décrit dans cette section le *modèle max*. Comme $\{t_{ar,1,j}\}_{1\leqslant j\leqslant n}$ sont indépendantes et qu'elles suivent la même loi de probabilité, la fonction de distribution de $t_{arg,1}$ se calcule ainsi [102] :

$$\begin{aligned} F_{t_{arg,1}}(x) &= P(\{t_{ar,1,1} \leqslant x\} \cap \{t_{ar,1,2} \leqslant x\} \cap \cdots \cap \{t_{ar,1,n} \leqslant x\}) \\ &= \prod_{i=1}^{n} P(\{t_{ar,1,j} \leqslant x\}) \text{ (indépendance des variables)} \\ &= (F_{t_{ar,1}}(x))^n \text{ (même loi de probabilité)} \end{aligned} \tag{6.24}$$

Soit $t_{arg,2}$ le temps d'aller-retours global nécessaire pour achever la seconde phase, de la même manière que pour la première phase, on a :

$$F_{t_{arg,2}}(x) = (F_{t_{ar,2}}(x))^n \tag{6.25}$$

On note enfin $t_{auth}$ le temps d'authentification, alors :

$$t_{auth} = t_{arg,1} + t_{arg,2} \tag{6.26}$$

Comme l'espérance mathématique, ou moyenne, est un opérateur linéaire [100], on a :

$$E(t_{auth}) = E(t_{arg,1}) + E(t_{arg,2}) \tag{6.27}$$

où $E(t_{auth})$ est l'espérance du temps total c.-à-d. la moyenne du temps d'authentification. Son expression et sa forme sont données dans la section suivante.

TABLEAU 6.2 – **Résumé des hypothèses**

1. Les nœuds ne se déplacent pas tout au long de la vie du réseau et les routes sont fixes.
2. Le nombre de pairs AAA est égal au seuil cryptographique : $n = th$.
3. Les temps de traitement des messages d'authentification sont négligeables.
4. Les nombres de sauts entre le JN et chaque pair AAA sont égaux.
5. La probabilité de retransmission est égale à 0.1 : $p = 0.1$

### 6.4.5 Résultats

Le temps $E(t_{auth})$ dépend de la longueur de chacun des messages d'authentification, du débit de données, de la technique d'étalement de spectre employée

par la couche physique, de la probabilité de retransmissions, du nombre de sauts et du nombre de pairs AAA. Pour les valeurs de ces paramètres résumées dans le tableau 6.3, nous avons fait varier le nombre de sauts et le nombre de pairs de la façon suivante :

- $n \in \{1, ..., 6\}$
- $hops \in \{1, ..., 10\}$

| paramètre | valeur |
|---|---|
| longueur du 1$^{er}$ message ($l_1$) | 287 octets |
| longueur du 2$^{ème}$ message ($l_2$) | 32 octets |
| longueur du 3$^{ème}$ message ($l_3$) | 1593 octets |
| longueur du 4$^{ème}$ message ($l_4$) | 1925 octets |
| débit ($\lambda$) | 11 Mbps |
| SIFS | 10 $\mu s$ |
| DIFS | 50 $\mu s$ |
| SlotTime ($\theta$) | 20 $\mu s$ |
| CWmin | 32 |
| ACK_Timeout | 334 $\mu s$ |
| longueur du message ACK | 304 bits |
| temps d'émission du message ACK ($EMd_{ACK}$) | 304 $\mu s$ |
| probabilité de retransmission ($p$) | 0.1 |

Tableau 6.3 – Valeurs des paramètres utilisés dans le modèle

La technique d'étalement de spectre que nous avons retenue est DSSS. Elle est utilisée par la norme IEEE 802.11. Les tailles des messages sont indiquées en fonction des contenus de ces derniers lorsqu'ils sont traités au niveau de la couche application (cf. section 4.4.4 du chapitre 4) et suivant l'exemple donné dans [103].

En prenant en compte les valeurs affichées au tableau 6.3, l'équation 6.27 devient :

$$\begin{aligned} E(t_{auth}) \quad = \quad & 866.73 \int_{-\infty}^{+\infty} \left( \frac{1}{2} - \frac{1}{2} \operatorname{erf} \left( \frac{0.61\, 10^{-8}(-0.25\ 10^{12} t + 0.29\ 10^{9} hops)}{\sqrt{hops}} \right) \right)^{(n-1)} \\ & \times \exp \left( - \frac{0.38\ 10^{-16}(-0.25\ 10^{12} t + 0.29\ 10^{9} hops)^2}{hops} \right) \frac{nt}{\sqrt{hops}} dt \\ + \ & 385.82 \int_{-\infty}^{+\infty} \left( \frac{1}{2} - \frac{1}{2} \operatorname{erf} \left( \frac{0.34\ 10^{-8}(-0.2\ 10^{12}\ t + 0.75\ 10^{9}\ hops)}{\sqrt{hops}} \right) \right)^{(n-1)} \\ & \times \exp \left( - \frac{0.12\ 10^{-16}(-0.2\ 10^{12} t + 0.75\ 10^{9} hops)^2}{hops} \right) \frac{nt}{\sqrt{hops}}\ dt \end{aligned} \tag{6.28}$$

(MAPLE présente ses calculs avec une précision de l'ordre de $10^{10}$, nous n'avons retenu que deux décimales significatives (cf. section E.1 de l'annexe E).)

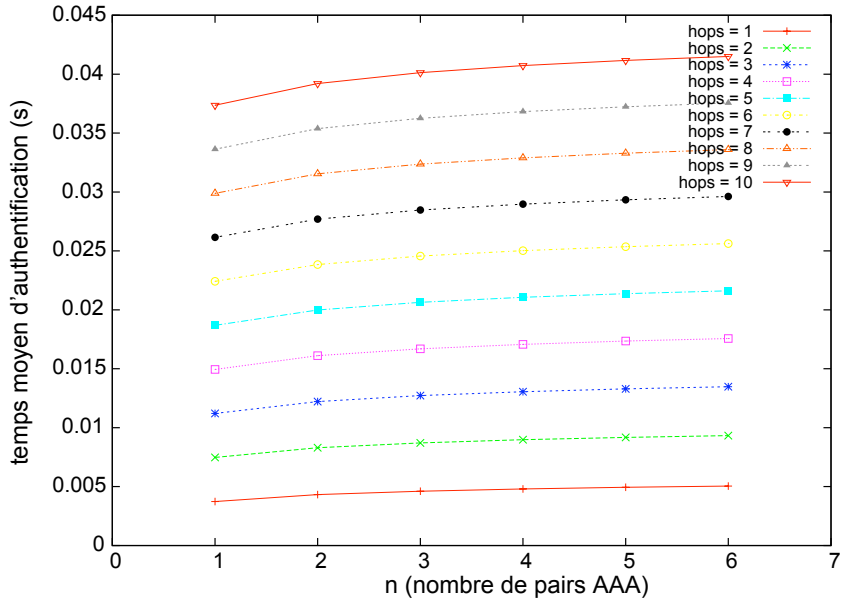


FIGURE 6.6 – Temps moyen d'authentification dans le modèle *max*

La figure 6.6 représente l'évolution du temps d'authentification $E(t_{auth})$ en fonction du nombre de pairs AAA et du nombre de sauts. Comme on pouvait s'y attendre, $E(t_{auth})$ croît lorsque $n$ croît et aussi lorsque *hops* croît. Les résultats sont compris entre 0.003 s pour $n = 1$ et $hops = 1$ et 0.04 s pour $n = 6$ et $hops = 10$, ce qui est largement acceptable.

Cependant ces résultats ne peuvent pas être admis tels quels au vu des hypothèses simplificatrices qui ont été choisies. On constate très vite, pourtant, qu'en raison de la complexité des calculs, il devient difficile de formaliser plus finement un tel système avec des méthodes analytiques. Une méthode de confirmation est nécessaire.

## 6.5 Modèles emboîtés et simulations

Dans cette section, nous utilisons le simulateur NS-2 [104] afin de confirmer d'abord les résultats obtenus grâce au modèle mathématique réalisé dans la section précédente. La collection des modèles intégrés dans NS-2 nous permettra ensuite d'aller plus loin en travaillant avec un modèle prenant en compte le mouvement des nœuds.

### 6.5.1 Considérations techniques, paramétrages et conditions de validation

NS-2 est un simulateur de réseaux à événements discrets. Comportant dès sa conception la plupart des protocoles courants, il est extensible et permet ainsi l'intégration de nouveaux protocoles. Il offre la possibilité de configurer divers scénarios de réseaux sans-fil et permet de simuler un réseau de quelques dizaines à quelques centaines de nœuds. Le cœur de NS-2 est écrit en C++ et le langage OTCL (*Object Tool Command Language*) [105] permet de spécifier les scénarios.

Le simulateur NS-2 intègre des éléments stochastiques sur lesquels nous nous étendrons pas dans ce chapitre. Cela tient à plusieurs raisons : d'une part les simulations auxquelles nous aurons recours ne reposent pas ou peu sur le hasard. Dès lors le simulateur peut nous apparaître comme une simple boîte noire délivrant des mesures successives. D'autre part, le traitement de telles séries de mesures peut être conçu comme le traitement de mesures en provenance de n'importe quelle expérimentation.

Dans le chapitre suivant les simulations conduites sont nettement plus complexes, nous nous interrogerons à leur propos beaucoup plus en profondeur sur le choix, la mise en œuvre, le paramétrage du simulateur et nous conduirons alors une étude critique approfondie qui n'a pas sa place dans le présent chapitre.

Nous avons conduit les simulations sur la version NS-2.34 [104]. Outre l'écriture en C++ des modules gérant le protocole d'authentification et leur intégration dans NS-2 afin de lui conférer cette nouvelle fonctionnalité, travail sur lequel nous reviendrons par la suite, nous avons principalement écrit deux scripts OTCL et, en AWK (*Aho, Weinberger et Kernighan*) ou Python [106], quelques dizaines de filtres destinés à analyser les traces de simulations.

Le premier script OTCL met en place un réseau où les nœuds sont fixes. Les routes y sont donc établies dès la deuxième authentification et ne varient plus. Si l'on néglige la première authentification, on se trouve en effet dans la situation décrite dans le modèle de la section 6.4. Ce premier scénario a permis de valider le modèle simplifié. Le second de ces scripts met en place un scénario où les nœuds sont mis en mouvement selon une trajectoire bien choisie afin de maîtriser les changements de routes et éviter autant que possible les ruptures de connexité. Ce scénario nous a permis d'approfondir notre étude en évaluant l'impact du mouvement sur le temps d'authentification.

Par ailleurs, un programme écrit en C permet, pour chaque couple de valeurs ($n$, $hops$), de calculer les positions des nœuds dans le premier scénario. La figure 6.7 illustre leur positionnement pour $n = 3$ et $hops = 3$. Les nœuds sont placés sur des cercles concentriques dont le centre est un nœud arrivant JN. Les pairs AAA sont placés sur le cercle extérieur de rayon $100 \cdot hops$ mètres de telle façon que les distances entre tout couple de pairs AAA successifs $\mathrm{srv_j}$ et $\mathrm{srv_{j+1}}$ soient égales. Les rayons des autres cercles sont dans l'ensemble $\{100, ...., 100 \cdot (hops -$

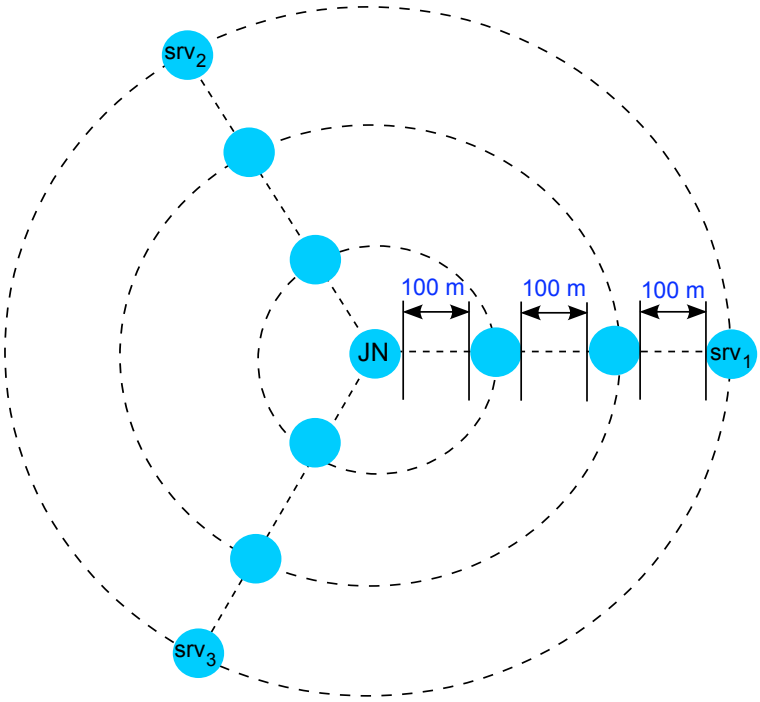


FIGURE 6.7 – Placement des nœuds dans le plan de simulation ($n = 3$, $hops = 3$)

1)}. Aux intersections de ces cercles avec les droites joignant le centre aux pairs AAA se trouvent les nœuds relais. Pour compléter l'étude en environnement mobile, un second programme écrit en C permet de calculer en plus des positions des nœuds, leurs trajectoires (cf. section 6.5.3).

Nous avons aussi écrit un script OTCL pour générer automatiquement le trafic d'authentification entre JN et le service AAA. JN lance une nouvelle authentification toutes les 10 s et cela 100 fois. Le trafic d'authentification est un trafic de type UDP. Les longueurs des messages sont celles données par le tableau 6.3. Les filtres décrits ci-dessus ont permis d'analyser les fichiers de trace produits habituellement par NS ainsi que d'autres, plus spécifiques dont nous avons programmé la sortie.

Ces simulations prenant un temps considérable, un script SHELL permet de les lancer automatiquement : un lancé de NS pour chaque couple ($n$, $hops$). Le tableau 6.4 donne les commandes NS utilisées dans les scripts OTCL et spécifie parmi les paramètres ceux qui nous ont paru les plus pertinents. Pour éviter que les résultats soient influencés par la longueur de la file d'attente de chaque nœud, sa valeur a été choisie suffisamment grande, soit 1000 paquets par file d'attente.

### 6.5.2 Scénario statique

Dans ce scénario, les nœuds sont statiques et les routes sont fixes durant toute la durée des simulations. Par construction, une route de JN à un pair AAA comporte tous les nœuds relais qui se trouvent à l'intersection des cercles

| paramètres et modèles | commandes NS | valeurs |
|---|---|---|
| protocole de routage | $ns_ node-config -adhocRouting | AODV |
| protocole MAC | $ns_ node-config -macType | Mac/802_11 |
| type de file | $ns_ node-config -ifqType | Queue/DropTail/PriQueue |
| longueur de file | $ns_ node-config -ifqLen | 1000 (paquets) |
| type d'antenne | $ns_ node-config -antType | Antenna/OmniAntenna |
| modèle de propagation | $ns_ node-config -propType | Propagation/FreeSpace [104] |
| portée du signal | Phy/WirelessPhy set RXThresh_ 8.5457e-09 | 150 (mètres) |
| technique DCF | Mac/802_11 set RTSThreshold_ 3000 | accès de base |
| technique d'étalement de spectre | (default NS model) | DSSS |
| débit | Mac/802_11 set dataRate_ | 11Mbps |
| coordonnées des nœuds | set X_, set Y_ | (dépend de $n$ et de *hops*) |
| durée d'une simulation ($d$) | $ns_ at $d "$ns_ halt" | 600 (secondes) |

Tableau 6.4 – Commandes NS et paramètres de simulations

concentriques et de la droite joignant JN à ce pair (cf. Figure 6.7). Nous avons exécuté NS pour chaque couple ($n$, *hops*) et obtenu les résultats illustrés par la figure 6.8.

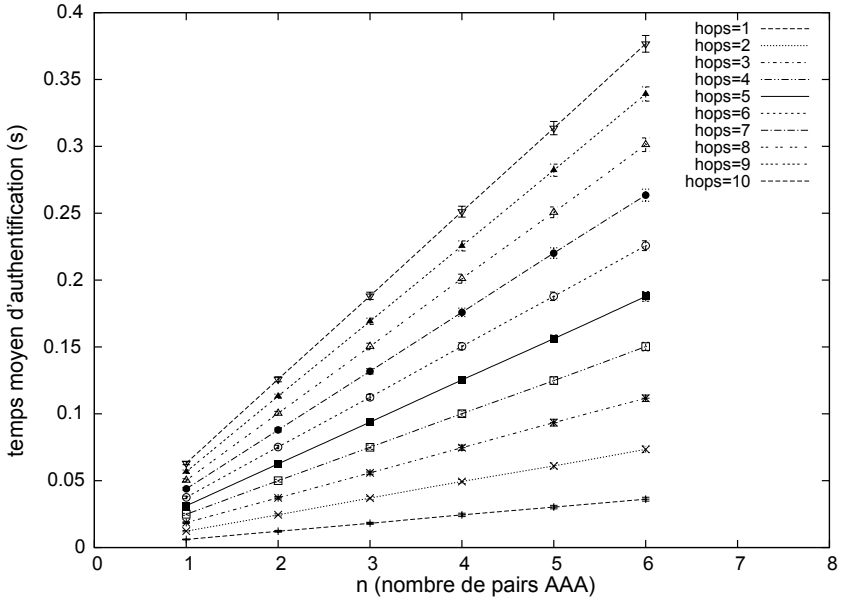


Figure 6.8 – Temps moyen d'authentification dans le scénario statique

Les courbes sont croissantes comme celles que nous avons obtenues par le calcul à la section 6.4. La longueur des intervalles de confiance est insignifiante [2]. En revanche, leur forme et la plage de valeurs globale sont significativement différentes. Alors que la forme des courbes est presque horizontale dans le modèle et qu'elles sont quasiment parallèles les unes aux autres, elles sont plutôt droites et de pentes croissantes en fonction du paramètre *hops* dans les simulations. La plage de valeurs est dix fois plus grande dans les résultats de simulations. On note, néanmoins, que pour les couples $\{(n = 1, hops = i)\}_{1 \leqslant i \leqslant 10}$, la plage de valeurs, à savoir $[0.006, 0.06]$, est assez proche de celle du modèle, à savoir $[0.003, 0.04]$. L'écart entre les plages de valeurs pour les autres valeurs de $n$ augmente en fonction de $n$.

L'analyse des fichiers traces produits par NS montre que le JN n'émet le message destiné à $\mathrm{srv_{j+1}}$ qu'une fois il a fini d'émettre le message destiné à $\mathrm{srv_j}$. Cela est dû au fait que les messages d'une requête donnée sont des messages *unicast*. Une requête vers $n$ pairs AAA est donc considérée comme $n$ messages différents. Par conséquent les temps d'émission de ces messages par le JN s'additionnent. Comme, en plus, le temps de propagation est négligeable par rapport au temps d'émission, les temps de transmission sont pratiquement réduits aux temps d'émission au niveau de tous les nœuds. Les distances entre les nœuds étant en plus égales, on s'aperçoit que les temps d'aller-retour, pour un échange donné, entre le JN et chaque pair s'additionnent.

Le modèle a été modifié pour prendre en considération cet aspect et nous avons obtenu les résultats de la figure 6.9. Dans le modèle modifié (cf. section E.2 de l'annexe E), au lieu de calculer le maximum des $\{t_{ar,1,j}\}_{1 \leqslant j \leqslant n}$, on a calculé leur somme. De ce fait, nous appelons ce modèle le *modèle somme*. Ainsi le temps d'aller-retours global $t_{arg,1}$ nécessaire à l'achèvement de la première phase est :

$$t_{arg,1} = \sum_{j=1}^{n} t_{ar,1,j} \tag{6.29}$$

Comme $\{t_{ar,1,j}\}_{1 \leqslant j \leqslant n}$ suivent la même loi de probabilité et que l'espérance mathématique est un opérateur linéaire [100] :

$$\begin{aligned} E(t_{arg,1}) &= E(\sum_{j=1}^{n} t_{rsf,1,j}) + E(\sum_{j=1}^{n} t_{rsf,2,j}) \\ &= n \cdot hops \cdot (\mu_{t_{rsm,1,j}} + \mu_{t_{rsm,2,j}}) \text{ (indépendant de j, cf. section 6.4.4)} \\ &= n \cdot \mu_{t_{ar,1}} \end{aligned} \tag{6.30}$$

De même, le temps d'aller-retours global $t_{arg,2}$ nécessaire pour achever la

2. La longueur de l'intervalle de confiance que l'on a considérée est égale à deux fois l'écart type, cela correspond à des estimateurs approximativement gaussiens. Mais en fait la valeur infime de ces écarts types donne des intervalles de confiance $[\mu - 4.47\sigma, \mu + 4.47\sigma]$ très réduits quelle que soit la distribution des estimateurs, en raison de l'inégalité de Biénaymé-Tchebytchev.

seconde phase vérifie :

$$\begin{aligned} E(t_{arg,2}) &= E(\sum_{j=1}^{n} t_{rsf,3,j}) + E(\sum_{j=1}^{n} t_{rsf,4,j}) \\ &= n \cdot hops \cdot (\mu_{t_{rsm,3,j}} + \mu_{t_{rsm,4,j}}) \text{ (indépendant de j} \\ &= n \cdot \mu_{t_{ar,2}} \end{aligned} \tag{6.31}$$

Ainsi, le temps total pour achever une authentification est :

$$E(t_{auth}) = E(t_{arg,1}) + E(t_{arg,1}) \tag{6.32}$$

$$\begin{aligned} &= n \cdot hops \cdot (\mu_{t_{rsm,1}} + \mu_{t_{rsm,2}} + \mu_{t_{rsm,3}} + \mu_{t_{rsm,4}}) \\ &= n\ hops\ (4\ p^3 ACK_Timeout\ \lambda + 4\ p\ ACK_Timeout\ \lambda\ + 2\ p\ CW_{min}\ \lambda \\ &\quad + 4\ p\ DIFS\ \lambda + 4\ p^5\ ACK_Timeout\ \lambda + 4\ p^7 ACK_Timeout\ \lambda \\ &\quad + 128\ p^7\ CW_{min}\ \lambda + 4\ p^4 ACK_Timeout\ \lambda + 16\ p^4\ CWmin\ \lambda \\ &\quad + 4\ p^4\ DIFS\ \lambda + 32\ p^5\ CWmin\ \lambda + 4\ p^5\ DIFS\ \lambda \\ &\quad + 4\ p^2\ ACK_Timeout\ \lambda + 4\ p^2\ CWmin\ \lambda + 4\ p^2\ DIFS\ \lambda \\ &\quad + 4\ p^6\ DIFS\ \lambda + 4\ p^6\ ACK_Timeout\ \lambda + 64\ p^6 CWmin\ \lambda \\ &\quad + 8\ p^3\ CWmin\ \lambda + 4\ p^3\ DIFS\ \lambda + 4\ p^7\ DIFS\ \lambda + 4\ \lambda\ SIFS + l_4 \\ &\quad + l_2 + l_1 + l_3 + 4\ \lambda\ DIFS + 4\ \lambda\ Ed_{ACK} + p^3\ l_3 + p^6\ l_3 + p^4\ l_3 \\ &\quad + p^5\ l_4 + p^3\ l_4 + p^4\ l_4 + p^6\ l_4 + p^7\ l_4\tilde{} + p^4\ l_2 + p\ l_2 + p^2\ l_3 \\ &\quad + p^5\ l_3 + p^7\ l_3 + p^6\ l_2 + p\ l_4\tilde{} + p\ l_3 + p^2\ l_4 + p^5\ l_2 + p^7\ l_2 \\ &\quad + p^3\ l_2 + p^2\ l_1 + p^5\ l_1 + p^7\ l_1 + p^3\ l_1 + p^4\ l_1 + p^6\ l_1 + p\ l_1 \\ &\quad + p^2\ l_2)/\lambda \end{aligned} \tag{6.33}$$

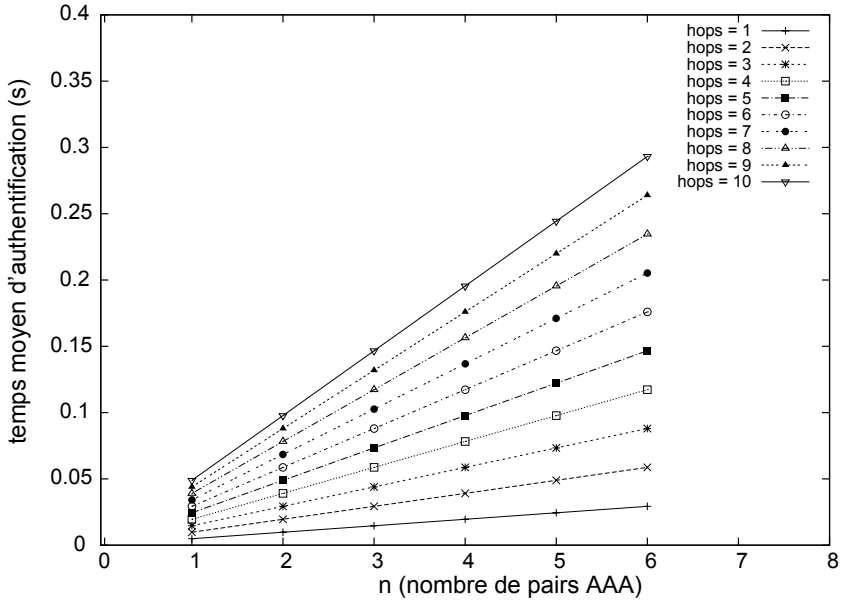


FIGURE 6.9 – Temps d'authentification moyen dans le modèle *somme*

La similarité entre les deux figures 6.8 et 6.9 est très grande. La plage de valeurs est un peu plus grande dans les simulations que dans la modélisation. La plus grande différence de valeurs est d'environ 0.08 s pour $n = 6$ et $hops = 10$. Elle est due aux temps de transmission et de réception entre les différentes couches dans les nœuds. Dans le modèle, ces temps étaient considérés comme négligeables. Ceci a été confirmé par les simulations dont les résultats affichent moins de 0.08 sec.

Nous en déduisons que les résultats de la modélisation et des simulations concordent. Cela nous conforte dans l'idée de pertinence des résultats.

### 6.5.3 Ajout de mouvements déterministes

Dans ce scénario, les routes sont initialement établies et les nœuds occupent les mêmes positions que dans le scénario précédent jusqu'à l'instant $t_0 = 300sec$. A $t_0 = 300sec$, les nœuds qui sont à un seul saut du JN sont mis en mouvement dans le sens trigonométrique. Au même instant, les nœuds qui sont à deux sauts du JN sont mis en mouvement dans le sens inverse. Chacun de ces nœuds se déplace linéairement à une vitesse de 5m/s vers le nœud suivant se trouvant sur le même cercle que lui même et dans le sens de mouvement choisi.

Les trajectoires ont été pensées de telle sorte que les nœuds du premier et du second sauts commencent à s'éloigner les uns des autres sans se déconnecter définitivement du réseau. A la fin des déplacements, ils se retrouvent placés aux intersections définies précédemment (cf. section 6.5.2).

Au cours du mouvement et lorsque des authentifications sont lancées, les routes sont établies ou calculées à nouveau en cas de changement en employant le protocole de routage AODV [12] (cf. annexe A). Certains nœuds se retrouvent parfois au-delà de la portée du reste des nœuds et deviennent ainsi injoignables. Cela peut arriver si $n \geqslant 3$ et $hops \geqslant 3$. Pour $n < 3$ et quelle que soit la valeur de $hops$, tous les nœuds sont situés sur une même droite, donc les nœuds qui entrent en mouvement ne deviennent jamais injoignables. Lorsque $n = 1$, on a un nœud unique au premier saut et de même au second saut si $hops \geqslant 2$. Lorsque $n = 2$, il y a deux nœuds au premier saut et de même au second saut si $hops \geqslant 2$. Ils sont tous alignés. Chacun des deux nœuds appartenant à un saut se déplace alors vers l'emplacement initial de l'autre nœud, donc il n'y a jamais de rupture de connectivité.

D'autre part, étant donné $hops < 3$ et quelle que soit la valeur de $n$, le graphe reste toujours connexe et les nœuds ne se déconnectent donc jamais du réseau. En effet, cela est vrai pour $n \in \{1, 2\}$ comme décrit dans le paragraphe ci-dessus. Le cas $n = 3$ peut être prouvé grâce à un calcul des distances entre les nœuds ainsi qu'il est décrit ci-dessous. Une fois cela montré pour $n \in \{1, 2, 3\}$, cela reste vrai pour $n > 3$ car les angles entre les lignes concourantes en JN et passant par les pairs AAA ne cessent de diminuer lorsque $n$ augmente.

La figure 6.10 illustre le cas où $n = 3$ et $hops = 3$. Les nœuds 31, 32 et 33 y désignent les pairs AAA. Au premier saut, les nœuds 11, 12 et 13 se déplacent respectivement vers 12, 13 et 11. Au deuxième saut, les nœuds 21, 22 et 23 se déplacent respectivement vers 23, 21 et 22. Les nœuds se trouvant au delà du deuxième saut, ne se déplacent pas.

Au cours du mouvement, il est clair que le nœud JN ne se déconnecte jamais des nœuds 11, 12, et 13 car ces derniers restent toujours inscrits dans le cercle de centre JN et de rayon 100 mètres (cf. table 6.4). Le calcul des distances séparant

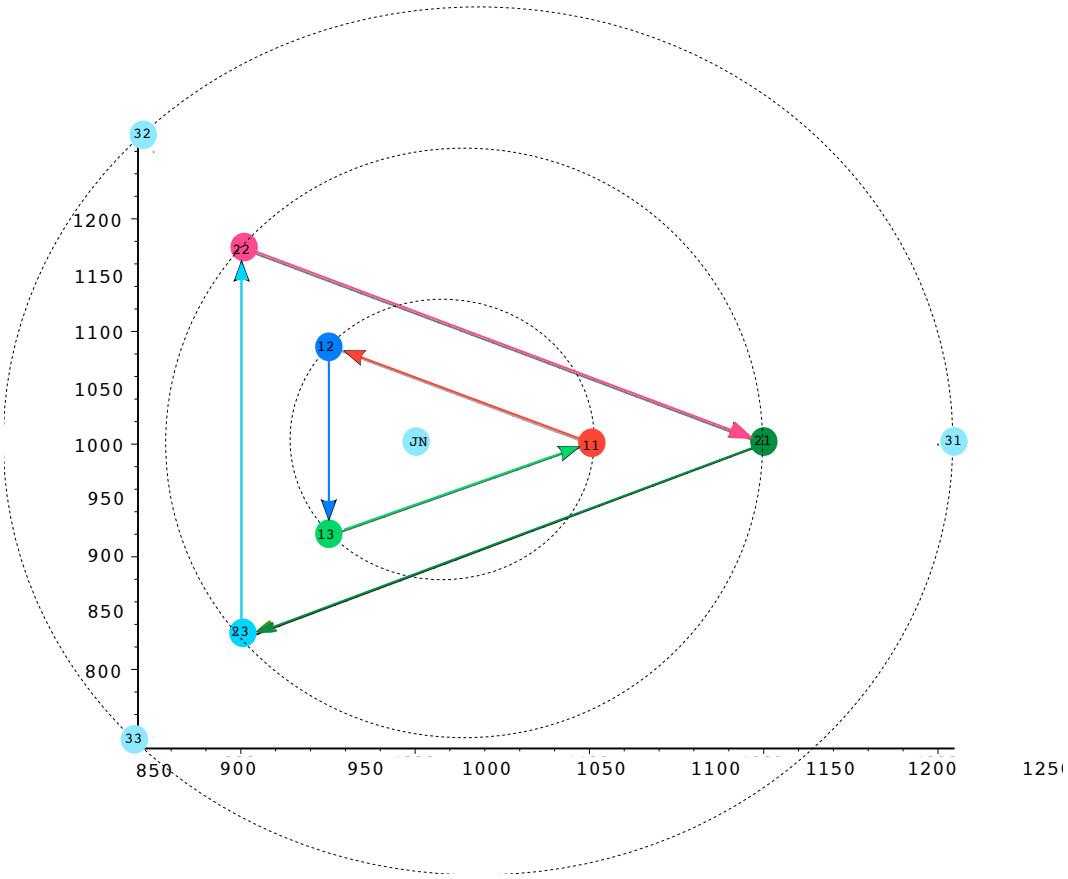


FIGURE 6.10 – Trajectoire des nœuds relais

le nœud 21 des nœuds 11, 12 et 13 démontre que celui-ci ne se déconnecte jamais de ces nœuds donc du réseau. Comme le montre la figure 6.11, la distance minimale entre le nœud 21 et les nœuds 11, 12 ou 13 est toujours inférieure à 150 mètres c.-à-d. la portée maximale du signal. Entre la date égale à 300 s et la date égale à 314 sec, le nœud 11 est le plus proche de 21. Entre les dates 315 s et 334 sec, le nœud 13 devient le plus proche de 21. Enfin, entre les dates 335 s et 368 s, le nœud 12 devient le plus proche de 21. A chaque fois que le nœud le plus proche de 21 changeait, nous avons observé qu'il était rapidement choisi par AODV pour devenir le nœud relais entre le JN et le nœud 21. Pour des raisons de symétrie, ce raisonnement reste valable pour les nœuds 22 et 23.

D'autre part, l'étude de l'évolution de la distance entre le nœud 31 et les nœuds 21 et 22, illustrée par le côté droit de la figure 6.11 montre que 31 se déconnecte du réseau entre les instants 312 s et 357 sec. Avant la date 311 sec, le relais des messages entre le JN et le nœud 31 est assuré par le nœud 21. Après la date 358 sec, il est assuré par le nœud 22. Pour des raisons de symétrie, ce raisonnement reste valable pour les nœuds 32 et 33. Ainsi, pour $n = 3$ et $hops = 3$, toute authentification lancée entre les dates 312 s et 357 s ne peut qu'échouer parce que le graphe n'est plus connexe dans cette plage là.

Notons, par ailleurs, que plus $n$ est grand, plus l'intervalle de temps où le

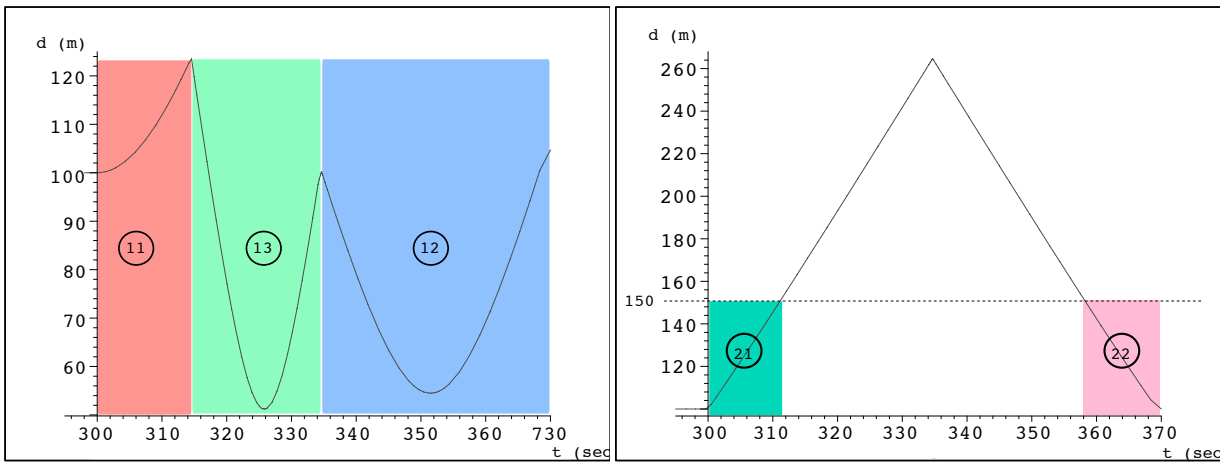


FIGURE 6.11 – Influence du mouvement des nœuds relais sur la connectivité du graphe du réseau. A gauche : la distance minimale entre le nœud 21 et les nœuds 11, 12 et 13. A droite : la distance minimale entre le nœud 31 et les nœuds 21 et 22

réseau demeure non connexe est court. Notons aussi que l'état non connexe n'intervient que pour $hops \geqslant 3$ et $n \geqslant 3$.

**Performance du protocole** Durant les simulations, pour chaque couple de valeurs ($n$, $hops$) où $n \in \{1..6\}$ et $hops \in \{1..10\}$ et à chaque seconde entre les moments du début et de la fin du mouvement, une authentification a été lancée par le JN. A la fin des simulations, nous avons calculé le taux de réussite $\tau$ défini comme le nombre d'authentifications ayant réussi divisé par le nombre total d'authentifications :

$$\tau = \frac{nombre\ d'authentifications\ abouties}{nombre\ d'authentifications\ lancées}$$

Le taux de succès enregistré est de 0.69 soit 69% des authentifications lancées qui ont abouti. L'examen des fichiers de traces (en anglais *trace files*), produits par le simulateur après chaque exécution, montre que le non aboutissement d'une authentification est souvent dû à la déconnexion du graphe comme expliqué ci-dessus. Dans ce cas les messages d'authentification sont supprimés par le nœud source. Un message d'erreur du type **NRTE : drop, no route is available** est alors écrit dans le fichier de traces selon le format suivant :

**d** -t 319.000000000 -Hs 0 -Hd -2 -Ni 0 -Nx 1000.00 -Ny 1000.00 -Nz 0.00 -Ne -1.000000 -Nl RTR -Nw **NRTE** -Ma 0 -Md 0 -Ms 0 -Mt 0 **-Is 0.2 -Id 3.0** -It udp -Il 347 -If 0 -Ii 0 -Iv 30 -Pn udp -Ps 3 -Pa 0 -Pf 0 -Po 0

où $d$ désigne un message supprimé (*drop*), l'option $-Is$ avec le paramètre $0.i$ identifie dans nos simulations le JN et l'option $-Id$ avec un paramètre $j.0$ identifie un pair AAA.

Les authentifications non abouties peuvent aussi avoir pour cause des collisions. Le message d'authentification est alors supprimé et tracé sous forme

**COL : drop, collision** comme ici :

**d** -t 339.094004956 -Hs 7 -Hd 7 -Ni 7 -Nx 919.10 -Ny 1058.78 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw **COL** -Ma 13a -Md 7 -Ms 0 -Mt 800 **-Is 0.1 -Id 2.0** -It udp -Il 1711 -If 0 -Ii 0 -Iv 30 -Pn udp -Ps 4 -Pa 0 -Pf 0 -Po 0

Lorsque le nombre de retransmissions d'un message d'authentification atteint sept (cf. annexe D), celui-ci est aussi supprimé, comme prévu par le standard IEEE 802.11 802.11, et tracé par un enregistrement **RET : drop, retry count exceeded** ainsi que le montre l'exemple suivant :

**s** -t 339.098445441 -Hs 7 -Hd 13 -Ni 7 -Nx 919.10 -Ny 1058.78 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw — -Ma 13a -Md d -Ms 7 -Mt 800 **-Is 0.1 -Id 2.0** -It udp -Il 1711 -If 0 -Ii 0 -Iv 29 -Pn udp -Ps 4 -Pa 0 -Pf 1 -Po 0

**s** -t 339.126988442 -Hs 7 -Hd 13 -Ni 7 -Nx 919.10 -Ny 1058.78 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw — -Ma 13a -Md d -Ms 7 -Mt 800 **-Is 0.1 -Id 2.0** -It udp -Il 1711 -If 0 -Ii 0 -Iv 29 -Pn udp -Ps 4 -Pa 0 -Pf 1 -Po 0

**s** -t 339.185584239 -Hs 7 -Hd 13 -Ni 7 -Nx 919.10 -Ny 1058.78 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw — -Ma 13a -Md d -Ms 7 -Mt 800 **-Is 0.1 -Id 2.0** -It udp -Il 1711 -If 0 -Ii 0 -Iv 29 -Pn udp -Ps 4 -Pa 0 -Pf 1 -Po 0

**s** -t 339.187391148 -Hs 7 -Hd 13 -Ni 7 -Nx 919.10 -Ny 1058.78 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw — -Ma 13a -Md d -Ms 7 -Mt 800 **-Is 0.1 -Id 2.0** -It udp -Il 1711 -If 0 -Ii 0 -Iv 29 -Pn udp -Ps 4 -Pa 0 -Pf 1 -Po 0

**s** -t 339.221528837 -Hs 7 -Hd 13 -Ni 7 -Nx 919.10 -Ny 1058.78 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw — -Ma 13a -Md d -Ms 7 -Mt 800 **-Is 0.1 -Id 2.0** -It udp -Il 1711 -If 0 -Ii 0 -Iv 29 -Pn udp -Ps 4 -Pa 0 -Pf 1 -Po 0

**s** -t 339.234775746 -Hs 7 -Hd 13 -Ni 7 -Nx 919.10 -Ny 1058.78 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw — -Ma 13a -Md d -Ms 7 -Mt 800 **-Is 0.1 -Id 2.0** -It udp -Il 1711 -If 0 -Ii 0 -Iv 29 -Pn udp -Ps 4 -Pa 0 -Pf 1 -Po 0

**s** -t 339.243102655 -Hs 7 -Hd 13 -Ni 7 -Nx 919.10 -Ny 1058.78 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw — -Ma 13a -Md d -Ms 7 -Mt 800 **-Is 0.1 -Id 2.0** -It udp -Il 1711 -If 0 -Ii 0 -Iv 29 -Pn udp -Ps 4 -Pa 0 -Pf 1 -Po 0

**d** -t 339.244839564 -Hs 7 -Hd 13 -Ni 7 -Nx 919.10 -Ny 1058.78 -Nz 0.00 -Ne -1.000000 -Nl MAC -Nw **RET** -Ma 13a -Md d -Ms 7 -Mt 800 **-Is 0.1 -Id 2.0** -It udp -Il 1653 -If 0 -Ii 0 -Iv 29 -Pn udp -Ps 4 -Pa 0 -Pf 1 -Po 0

**s** désigne un message qui a été envoyé.

Seulement pour les authentifications qui ont abouti, nous avons calculé le temps d'authentification puis le rapport $\rho$ pour chaque couple de valeurs ($n$,$hops$) où $\rho$ est défini par :

$$\rho = \frac{temps\ d'authentification\ dans\ le\ scénario\ sans\ mouvement}{temps\ d'authentification\ dans\ le\ scénario\ avec\ mouvement}$$

La valeur moyenne que nous avons obtenue pour $\rho$ est 0.23, ce qui signifie qu'une authentification aboutie dans le scénario avec mouvement nécessite approximativement quatre fois plus de temps qu'une authentification dans le cas statique. Le temps induit par le routage est donc trois fois plus important que le temps dû à l'authentification seule.

Les valeurs de $\tau$ et de $\rho$ démontrent que l'aboutissement d'une authentification et le temps qu'elle nécessite pour être achevée sont conditionnés par le processus de routage. Cependant quand une authentification aboutit, le temps de son exécution est inférieur à 1.4 sec.

### 6.5.4 Prise en compte des temps de calculs cryptographiques

Nous avons évalué le temps moyen nécessaire aux calculs cryptographiques lors de la construction et du traitement des messages d'authentification. Pour cela, nous avons écrit en langage C un programme utilisant la librairie OpenSSL. Nous l'avons exécuté sur une machine ayant une vitesse d'horloge de 2 GHz. Voici les temps moyens intermédiaires calculés :

- génération de nombre aléatoire : 0.000 ms
- génération de signature RSA : 9.883 ms
- vérification de signature RSA signature : 0.391 ms
- génération de jeton : 9.873 ms

Ces valeurs confirment que les temps de traitement sont négligeables ainsi que nous l'avons supposé précédemment.

## 6.6 Conclusion

Dans ce chapitre, nous avons formalisé le protocole d'authentification et utilisé des méthodes analytiques pour évaluer *grossièrement* le temps moyen d'authentification. Pour ce faire, nous avons simplifié autant que possible le modèle de réseau ad hoc et des couches de protocoles. Les résultats obtenus montrent que ce protocole peut être performant sous les hypothèses que nous avons définies (cf. tableau 6.2). Cela a fait l'objet d'un article [107] que nous avons présenté à la conférence *IEEE Wireless Communications, Networking and Information Security* (WCNIS) tenue en juin 2010 à Pékin.

Nous avons analysé le temps d'exécution du protocole d'authentification, défini au chapitre 4, entre un nœud arrivant et l'ensemble des pairs AAA. Le modèle simplifié que nous avons établi a montré que, lorsque les routes sont déjà établies et que les nœuds sont statiques au cours du temps, le temps d 'authentification croît lorsque le nombre de pairs augmente et, aussi, lorsque le nombre de sauts entre le nœud arrivant et chacun des pairs augmente. Sa valeur ne dépasse pas 300 millisecondes pour un maximum de 6 serveurs et de 10 sauts. Des simulations conduites sur NS-2 ont renforcé la validité de ces résultats.

Par ailleurs, nous avons créé un modèle de mouvement simple afin d'évaluer l'impact du routage sur le temps d'authentification total. Les simulations ont montré que pour les authentifications qui aboutissent ce temps est multiplié par quatre comparativement au scénario sans mouvement. Le temps de routage est donc trois fois plus important que le temps d'authentification lorsqu'il n'y a pas de mouvement. En moyenne, on peut s'attendre à ce qu'une authentification soit

terminée après 1.4 s pour un nombre de pairs AAA inférieur à six et un nombre de sauts inférieur à 10. Le protocole d'authentification peut donc être considéré comme performant et passant à l'échelle quand il y a du mouvement. Cela nous amène à penser que le temps d'authentification ne devrait pas représenter un frein au développement du protocole d'authentification que nous avons conçu. Néanmoins, une prise en compte des facteurs négligés dans cette partie, en particulier le caractère aléatoire du mouvement des nœuds et un nombre de pairs AAA différent du seuil cryptographique, est importante pour s'en assurer. Cela fera l'objet du chapitre suivant.

En ce qui concerne les authentifications qui n'ont pas abouti, le traitement de leur renouvellement a été prévu dans le chapitre 4 grâce à la définition de l'algorithme à multiples tentatives d'authentification. Les performances de celui-ci seront étudiées également dans le chapitre suivant.

# Chapitre 7

# Réglage fin et performances du protocole

## Sommaire


**7.1 Introduction . . . . . 150**
**7.2 Étude théorique : optimisation du taux de réussite par ajustement de seuils . . . . . 151**
7.2.1 Choix du seuil cryptographique *th* . . . . . 151
7.2.2 Introduction de seuils intermédiaires . . . . . 152
7.2.3 Choix de seuils intermédiaires optimaux . . . . . 152
**7.3 Choix du simulateur et des modèles . . . . . 155**
7.3.1 Simulation de réseaux : aperçu . . . . . 155
7.3.2 Choix du simulateur . . . . . 156
7.3.2.1 NS-3 . . . . . 158
7.3.2.2 OPNET . . . . . 158
7.3.2.3 OMNET++ . . . . . 159
7.3.2.4 NS-2 . . . . . 160
7.3.2.5 Notre choix . . . . . 162
7.3.3 Modèles utiles et classification . . . . . 162
7.3.3.1 Modèles de mouvement . . . . . 162
7.3.3.2 Modèles de propagation . . . . . 164
7.3.3.3 Modèles de trafic . . . . . 165
7.3.3.4 Nos choix . . . . . 165
**7.4 Travail préparatoire aux simulations dans NS-2 . 166**
7.4.1 Organisation des outils et des données . . . . . 166
7.4.2 Extension de NS-2 par une implémentation du protocole d'authentification . . . . . 168
7.4.2.1 Cahier des charges et simplification . . . . 168
7.4.2.2 Création de nouvelles classes . . . . . 169
7.4.2.3 En-tête AAA et numéro de séquence . . . . 169
7.4.2.4 Classe *UdpAAAagent* . . . . . 170
7.4.2.5 Classe *AAAClient* . . . . . 170
7.4.2.6 Classe *AAAServer* . . . . . 173

7.4.2.7 Traces AAA . . . . . 173
**7.5 Scénarios de simulations et résultats . . . . . 173**
7.5.1 Paramétrages généraux . . . . . 174
7.5.1.1 Mouvement . . . . . 174
7.5.1.2 Couche sans-fil et propagation . . . . . 174
7.5.1.3 Trafic ambiant . . . . . 174
7.5.2 Paramétrage du trafic AAA d'authentification . . . . 176
7.5.2.1 Paramétrage du protocole . . . . . 176
7.5.2.2 Paramétrage du trafic . . . . . 177
7.5.3 Scénarios . . . . . 178
7.5.4 Résultats . . . . . 179
**7.6 Réglages de l'algorithme à multiples tentatives d'authentification . . . . . 184**
7.6.1 Estimation du temps d'attente maximum $t_{max}$ . . . 184
7.6.1.1 Calcul de $t_{max}$ et validation de la probabilité de réussite . . . . . 184
7.6.2 Réglage du nombre maximum de tentatives $K$ . . . 185
**7.7 L'entrelacement des phases est-il une amélioration ? 185**
7.7.1 Contribution de l'approche d'*entrelacement* par rapport à l'approche de *séparation* . . . . . 186
**7.8 Conclusion . . . . . 187**

## 7.1 Introduction

Un des indicateurs de performance d'un protocole de réseau est sa capacité à se terminer avec une issue favorable, on dit aussi réussir. C'est un facteur déterminant dans l'adoption ou l'abandon de ce protocole. Pour les mêmes raisons qu'au chapitre 6, nous nous intéressons alors à la capacité de terminaison, ou *taux de réussite* du protocole d'authentification défini à la section 4.4.4 du chapitre 4. Les facteurs considérés dans cette étude sont liés à l'activité du réseau en termes de mouvement et de trafic et ne s'intéressent pas aux capacités cryptographiques des nœuds. On suppose donc que ceux-ci ont suffisamment de ressources pour générer un contenu cryptographique d'un message d'authentification ou vérifier la validité de tels contenus. Les nœuds sont, de plus, en possession d'éléments d'accréditation valides et tout échec d'authentification ne peut survenir qu'en raison de conditions extérieures liées au réseau.

Au chapitre 6, nous avons construit un modèle rudimentaire du réseau MANET afin de déterminer le temps d'authentification. Une des hypothèses considérait que le seuil cryptographique était égal au nombre de pairs AAA présents dans le réseau. Nous verrons dans ce chapitre que cette hypothèse cesse d'être adaptée si l'on veut maximiser le taux de réussite en tenant compte des facteurs ci-dessus.

Une fois que les conditions requises pour obtenir un taux de réussite optimal ont été déterminées, une évaluation des paramètres de réglages de l'algorithme à tentatives multiples (cf. section 4.4.5 du chapitre 4), à savoir la durée d'attente maximale avant de relancer une nouvelle tentative et le nombre total de tentatives, est conduite. L'apport de cet algorithme dans le processus complet d'authentification est aussi mesuré. Quelques idées pouvant améliorer à la fois

le taux de réussite et les valeurs de ces paramètres sont, par ailleurs, présentées et discutées. L'ensemble de ces travaux a été présenté dans une publication à la conférence IEEE ISCC qui a eu lieu à Corfou en juin 2011 [84]. La simulation y joue un rôle particulièrement important. Pour cette raison, nous présentons une étude assez approfondie des simulateurs et de la simulation appliquée au domaine des réseaux MANETs.

## 7.2 Étude théorique : optimisation du taux de réussite par ajustement de deux seuils

Afin d'être authentifié d'abord et de recevoir ensuite son jeton d'accès, un nœud client arrivant, appelé JN (cf. tableau 4.2), a besoin de contacter au moins un nombre seuil $th$ de pairs AAA durant deux phases. L'acquisition du jeton est importante car il prouve l'autorisation du JN à accéder aux services souscrits.

Chaque phase du protocole d'authentification consiste en deux étapes (cf. définition 6.3.1 de la section 6.3 du chapitre 6). Lors de chaque étape, il est émis au moins $th$ messages *unicast*. S'il s'agit de la première ou de la troisième étape, ces messages sont des requêtes adressées par le JN aux pairs AAA. S'il s'agit de la deuxième ou de la quatrième étape, ces messages sont des réponses des pairs AAA aux requêtes du JN. Le processus d'authentification est considéré comme réussi si, à la deuxième étape, le JN reçoit au moins $th$ réponses d'un sous ensemble de pairs AAA et, si à la quatrième étape, il reçoit, également, au moins $th$ réponses de pairs appartenant au même sous ensemble précédent. Nous supposons que cela est suffisant pour considérer que l'issue d'une authentification est favorable ce qui signifie que le contenu de ces réponses est supposé toujours valide et favorable lui aussi.

Le paramètre $th$ provient de l'emploi, dans ce protocole, d'un système de cryptographie à seuil RSA de paramètres $(n, th)$. $n$ est le nombre total de pairs AAA et $th$ est le seuil cryptographique. La clé privée du système est partagée entre les $n$ pairs de telle sorte qu'un groupe quelconque d'au moins $th \geqslant 2$ pairs soit capable de *co-signer* une information [1].

### 7.2.1 Choix du seuil cryptographique $th$

Mis à part les détails rappelés ci-dessus, la spécification donnée à la section 4.4 du chapitre 4 précise qu'en raison de continuité de service, même lorsque $th-1$ pairs AAA sont compromis, on a l'inégalité : $th \leqslant \lfloor \frac{n+1}{2} \rfloor$. Nous supposons donc tout au long de ce chapitre, que :

$$2 \leqslant th \leqslant \lfloor \frac{n+1}{2} \rfloor \tag{7.1}$$

Quant au choix, en pratique, d'une valeur de $th$, un compromis est à faire entre la sécurité et la commodité du système. Augmenter la valeur du seuil cryptographique permet d'atteindre un meilleur niveau de sécurité puisque, afin de mettre le système en panne, un adversaire mobile [76] doit compromettre au moins $th$ pairs. Mais, en conséquence, le système devient incommode parce

1. Le cas $th = 1$ est celui où les pairs AAA possèdent tous la clé du système et sont donc tous capables de signer seuls. En ce sens, ils sont une réplique les uns des autres. Ce cas a été exclu de notre champ de travail comme montré à la section 4.2.1 du chapitre 4.

que la probabilité de recevoir $th$ messages aura diminué et le taux de réussite aussi. La charge du réseau aura aussi augmenté. Pour cela, dans ce chapitre, nous testons deux cas : le cas où $th$ est assez petit, mais pas plus petit que 3 pour que la cryptographie à seuil ait un sens, et le cas où $th$ est le plus grand possible c.-à-d. égal à $\lfloor \frac{n+1}{2} \rfloor$.

Néanmoins, cela étant fixé, il reste à préciser le nombre de pairs qu'un JN doit solliciter à la première et à la troisième étapes. Pour répondre à cette question, nous avons suivi un raisonnement basé sur l'évaluation du taux de réussite sachant que l'objectif est de le maximiser sans trop augmenter la charge du réseau.

### 7.2.2 Introduction de seuils intermédiaires

Le couple $(n, th)$ étant fixé, à la question combien de pairs AAA solliciter, $th$ est une réponse plausible si l'on souhaite minimiser la charge du réseau. Par contre, si l'on désire maximiser le taux de réussite, $n$ est une réponse évidente. Il est clair, par ailleurs, que le choix $th$ minimise le taux de réussite, alors que le choix $n$ maximise la charge du réseau. Il y a, donc, un compromis à faire. Une solution serait alors de choisir une valeur intermédiaire, que l'on note $th_M$, telle que : $th \leqslant th_M \leqslant n$. Un JN contactera ainsi $th_M$ pairs AAA lors d'une phase du protocole et attendra au moins $th$ messages en réponse.

Cette solution aurait pu être suffisante si l'on avait à traiter un protocole à une seule phase. Comme il y en a deux, on est amené à définir une valeur intermédiaire pour chacune d'entre elle. En plus de $th_M$, qui sera utilisée à la première phase, appelons alors $th_i$ celle qui sera utilisée dans la deuxième phase. Il est clair que $th \leqslant th_i \leqslant n$.

En reprenant les notations d'ensembles fixées à la section 4.4.3 du chapitre 4, les pairs sollicités par un JN pendant l'étape 1 (phase 1) sont dans un ensemble $\mathcal{P}_1$, donc $th_M = card(\mathcal{P}_1)$. Dans l'ensemble des pairs ayant répondu et dont les réponses sont parvenues au JN, seulement un sous ensemble $\mathcal{P}_4$ est sollicité à l'étape 3 (phase 2), donc $th_i = card(\mathcal{P}_4)$. Enfin, plusieurs pairs répondent et les parts de signature ainsi que les parts de jeton d'un sous ensemble d'entre eux, noté $\mathcal{P}_6$, sont combinés. On a donc $card(\mathcal{P}_6) = th$. A chaque phase, le nombre de pairs impliqués réduit soit parce que certains n'ont pas répondu car ils n'ont pas pu recevoir les sollicitations du JN, soit parce que, tout en ayant répondu, leurs réponses ne sont pas parvenues au JN. Les inclusions d'ensembles $(\mathcal{P}_6 \subseteq \mathcal{P}_4 \subseteq \mathcal{P}_1)$ permettent, par ailleurs, d'écrire : $th \leqslant th_i \leqslant th_M$.

En résumé, deux nombres intermédiaires $th_M$ et $th_i$ sont à définir tels que :

$$th \leqslant th_i \leqslant th_M \leqslant n \tag{7.2}$$

On les appelle les seuils intermédiaires.

Une étude analytique présentée ci-dessous nous permet de déterminer les valeurs des seuils $th_i$ et $th_M$ pour un taux de réussite optimal.

### 7.2.3 Choix de seuils intermédiaires optimaux

D'un point de vue purement théorique, nous supposons que chaque pair sollicité répond avec une probabilité $p$. Nous supposons aussi que les pairs agissent indépendamment les uns des autres.

Sans faire appel aux seuils intermédiaires, la probabilité qu'un nombre exact de $th$ pairs répondent est égale à $p^{th}$. C'est une fonction décroissante du seuil cryptographique $th$ comme nous l'avons déjà indiqué à la section précédente : la probabilité de réponse décroît lorsque la sécurité croît.

En utilisant les seuils intermédiaires $th_M$ et $th_i$ à la première phase, JN sollicite $th_M$ pairs et attend au moins $th_i$ messages en réponse. La probabilité de cet événement est :

$$P_1(th_i, th_M) = \sum_{j=th_i}^{th_M} \binom{th_M}{j} p^j (1-p)^{th_M - j} \tag{7.3}$$

De la même manière, au deuxième échange, JN sollicite $th_i$ pairs et attend au moins $th$ messages en réponse. Un calcul avec le logiciel MAPLE permet alors d'obtenir comme probabilité de finaliser une authentification l'expression suivante :

$$\begin{aligned} P_2(th, th_i, th_M) &= P_1(th, th_i) \cdot P_1(th_i, th_M) \\ &= \left( \sum_{j=th}^{th_i} \binom{th_i}{j} p^j (1-p)^{th_i - j} \right) \times \\ &\quad \left( \sum_{j=th_i}^{th_M} \binom{th_M}{j} p^j (1-p)^{th_M - j} \right) \end{aligned} \tag{7.4}$$

$P_2$ est ce que nous avons appelé le *taux de réussite*.

| Paramètre | Valeur |
|---|---|
| estimation de la probabilité de réponse d'un pair AAA | $\hat{p} = 0.85$ |
| nombre total de nœuds dans le réseau MANET | 100 |
| nombre total de pairs AAA | $n = 20$ |
| seuil cryptographique assez petit | $th = 5$ |
| seuil cryptographique assez grand | $th = 10$ |

Tableau 7.1 – Valeurs des paramètres

L'étude des variations de $P_2$ en fonction de $th_i$ pour différentes valeurs de $th_M$ nous permet de choisir, ensuite, des valeurs "convenables" de ces deux seuils (cf. figure 7.1).

Après exécution de plusieurs simulations, sous les mêmes conditions que celles définies à la section 7.5.4, nous avons estimé la probabilité $p$ qu'un pair réponde à $\hat{p} = 0.85$. Ce paramètre ainsi que les valeurs des paramètres $n$ et $th$ sont résumés dans le tableau 7.1. Nous avons pris $n = 20$ pairs AAA pour un nombre total de nœuds égal à 100, soit un cinquième, ce que nous estimons raisonnable. $th$=5 et $th$=10 sont les possibilités proposées à la section 7.2.1.

La partie gauche de la figure 7.1 montre, pour $th = 5$ et $th_M \in \{5, ..., 20\}$, les variations de $\widehat{P_2}$, l'estimation de $P_2$, en fonction de $th_i \in \{5, ..., th_M\}$. La partie

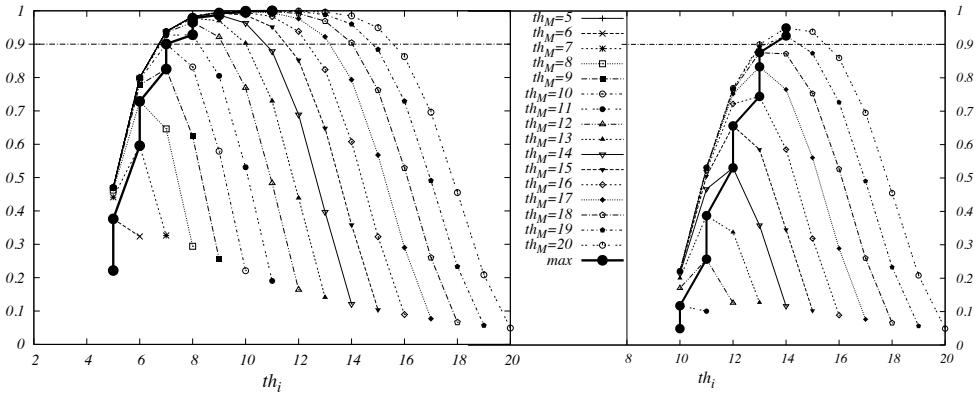


FIGURE 7.1 – A gauche : $\widehat{P_2}$ lorsque $th = 5$, $th_M \in \{5, ..., 20\}$. A droite : $\widehat{P_2}$ lorsque $th = 10$, $th_M \in \{10, ..., 20\}$

droite de cette figure montre pour $th = 10$ et $th_M \in \{10, ..., 20\}$ les variations de $\widehat{P_2}$ en fonction de $th_i \in \{10, ..., th_M\}$.

Lorsque $th = 5$, $\widehat{P_2}$ dépasse 90% pour $th_M \geqslant 10$ et $th_i \geqslant 7$. Bien qu'augmenter $th_M$ augmenterait davantage $\widehat{P_2}$, ce n'est pas intéressant de choisir un $th_M \geqslant 12$. En effet, pour ces valeurs de $th_M$, les valeurs maximales atteintes par $\widehat{P_2}$, illustrées sur la figure par la partie quasiment plate de la courbe à ligne continue et épaisse de légende *"max"*, n'augmentent plus significativement. Si on appelle *gain relatif* entre $x_i$ et $x_j$ tels que $x_i \leqslant x_j$, la quantité :

$$\frac{\widehat{P_2}(x_j) - \widehat{P_2}(x_i)}{\widehat{P_2}(x_i)} \tag{7.5}$$

On remarque que ce gain n'est pas significatif dans cette zone. En diminuant, il passe, d'à peu près 3% entre $th_M = 10$ et $th_M = 11$ à environ 0.04% entre $th_M = 19$ et $th_M = 20$. Ainsi, choisir une valeur de $th_M$ dans cette zone chargerait inutilement le réseau par des messages d'authentification supplémentaires. Nous choisissons donc $th_M = 11$ pour laquelle la courbe *"max"* donne $th_i = 8$.

De la même manière, lorsque $th = 10$, pour les valeurs maximales de $\widehat{P_2}$ illustrées par la courbe *"max"*, le gain relatif diminue au fur et à mesure que $th_M$ augmente. Il passe d'à peu près 5% entre $th_M = 18$ et $th_M = 19$ à environ 2.5% entre $th_M = 19$ et $th_M = 20$. Nous choisissons $th_M = 19$ pour laquelle la courbe *"max"* donne $th_i = 14$.

| Triplets de seuils | $\widehat{P_2}$ |
|---|---|
| ($th = 5$, $th_i = 8$, $th_M = 11$) | 0.93 |
| ($th = 10$, $th_i = 14$, $th_M = 19$) | 0.92 |

TABLEAU 7.2 – Récapitulatifs des valeurs optimales choisies et du taux de réussite correspondant

Le tableau 7.2 récapitule le choix des valeurs optimales des deux seuils, $th_i$

et $th_M$, introduits dans cette section. Il les présente sous forme de triplets avec la valeur du seuil cryptographique pour laquelle ils ont été fixés. En face de chaque triplet, le taux de réussite, lisible à la figure 7.1, a été indiqué.

Dans le reste de ce chapitre, nous allons mener des simulations prenant en compte ces triplets. Notre but est de déterminer les valeurs de certains autres paramètres nécessaires à l'implémentation du protocole d'authentification. Ces paramètres concernent la définition de l'algorithme à tentatives multiples d'authentification (cf. 4.4.5 du chapitre 4) et consistent en :

- $t_{max}$ : la durée d'attente maximale avant d'abandonner une tentative d'authentification non réussie et d'en lancer une autre.
- $K$ : le nombre maximum de tentatives à lancer à la suite desquelles on est *quasiment* sûr que l'authentification sera réussie.

Ces simulations confirmeront, par ailleurs, les valeurs du taux de réussite indiquées au tableau 7.2.

La section 7.7 étudiera, par la suite, en procédant aussi par des simulations, une proposition susceptible d'améliorer les valeurs de $t_{max}$, de $K$ et du taux de réussite.

Puisque la simulation sera un outil important dans ce qui suit, nous commençons par en présenter un aperçu dans la section suivante. En particulier, nous passons en revue plusieurs simulateurs de réseaux afin de justifier le choix de simulateur que nous avons opéré. Nous étudions aussi plusieurs modèles en général présents dans ces simulateurs et importants à connaître avant la mise en place de tout plan de simulation.

## 7.3 Choix du simulateur et des modèles

Si l'on veut utiliser le protocole d'authentification dans un réseau ad hoc réel, il est important d'évaluer ses performances dans un contexte aussi proche que possible de la réalité. Nous serons donc amenés à relâcher les hypothèses que nous avons émises au chapitre 6 pour la commodité des raisonnements et ainsi à oublier une grande partie des hypothèses simplificatrices de départ. La topologie du réseau va être construite sur un nombre de nœuds plus élevé. Un trafic représentatif des conditions réelles de fonctionnement devra être modélisé puis mis en place. On prendra désormais en compte les aléas introduits par la mobilité et aussi les aléas introduits par les transmissions. Tout cela nécessitera une modélisation adéquate, impliquera une variance supérieure des estimateurs et devra être solidement fondé sur des résultats statistiques bien établis.

On constate très vite qu'il devient difficile de mathématiser entièrement un tel système et que les méthodes analytiques sont impuissantes à la résolution explicite dans un modèle aussi complexe. Nous avons donc opté pour la méthode de simulation.

### 7.3.1 Simulation de réseaux : aperçu

La simulation permet de modéliser des objets ou des activités réels ou imaginaires afin de reproduire le fonctionnement d'un système et d'en mesurer les performances [108]. Elle peut être employée dans divers domaines d'applications. En particulier, la simulation assistée par ordinateur est utilisée en physique, chimie, biologie, économie, finance, gestion de ressources, ingénierie, etc.

Son emploi dans le domaine des réseaux remonte aux années 1960.

Il existe trois méthodes de simulation : soit par l'utilisation d'environnements spécialisés tels que SIMAN/ARENA [109] ; soit par des langages dédiés à la simulation tels que SIMULA [110], SIMAN [111] ou QNAP2 [112] ou des bibliothèques pour la simulation comme JAVASIM, C++SIM, etc ; soit, enfin, par l'utilisation de langages de programmation généraux tels que C, Java ou Delphi.

Comme la pile de protocoles et les éléments se rapportant aux réseaux ad hoc sont multiples et complexes à développer, nous avons choisi de ne suivre ni la deuxième, ni la troisième méthode et d'éviter ainsi de programmer tout *ex nihilo*. Cette décision se justifie aussi par l'existence d'environnements spécialisés dans la simulation des réseaux qui ont fait leurs preuves dans le domaine de la recherche et parfois aussi dans les mondes industriel et éducatif.

Les simulateurs de réseaux modélisent des réseaux réels. Leurs principaux modèles sont à événements discrets. Plus ils sont proches de la réalité, meilleure est la qualité de leurs résultats. Il est plus facile de modifier le modèle d'un protocole ou d'une technologie, développés pour travailler sur un simulateur donné, que de changer leurs implémentations dans un réseau réel. Dans le domaine des réseaux, les chercheurs universitaires, tout comme les développeurs dans l'industrie, font appel aux simulateurs pour étudier la faisabilité de nouveaux protocoles ou de nouvelles technologies, analyser leurs performances, ou évaluer celles de protocoles existants. Parfois il suffit pour ce faire de changer simplement les valeurs de quelques paramètres.

Les simulateurs de réseaux sont soumis à des contraintes liées à l'évolution rapide et continuelle des réseaux. Leurs plate-formes doivent, par exemple, offrir une intégration rapide, facile et transparente de nouveaux *packages*. Généralement, un simulateur de réseaux implante une large variété de protocoles et de technologies réseau. Il met aussi à disposition des utilisateurs des briques de base permettant de construire des réseaux de complexités diverses. Parmi ces briques se trouvent les nœuds fixes, les nœuds mobiles, les routeurs, les hubs, les stations de base, les liens filaires et sans-fil, etc.

Au minimum, un simulateur permet de représenter la topologie du réseau, de spécifier l'architecture des nœuds et de préciser le type de liens qui les relie. Il doit aussi permettre de décrire le trafic et de définir les scénarios. Des logiciels plus complexes ont la possibilité de spécifier en outre le détail des protocoles qui traitent le trafic.

### 7.3.2 Choix du simulateur

Le tableau 7.3 classe les simulateurs les plus connus selon qu'ils soient commerciaux ou gratuits avec un code source libre. Les simulateurs payants ont, en général, l'avantage d'être développés et maintenus par une équipe d'experts. Ils sont à jour des nouveaux standards et RFC et leur documentation est complète. Les simulateurs gratuits, par contre, ne le sont pas toujours. L'absence de documentation complète peut constituer un sérieux handicap surtout lorsque les nouvelles versions sont livrées avec des packages différents. Toutefois, grâce à leur flexibilité, les simulateurs libres sont plus ouverts aux modifications et aux contributions ; de plus, le caractère *Open source* en autorisant chaque chercheur à aller regarder dans le détail le fonctionnement des différents modules garantit la très haute qualité du code utilisé.

| Simulateur | Commercial | Gratuit / libre | Licence | Date de dernière version | Système d'exploitation | Développé par |
|---|---|---|---|---|---|---|
| NS-2 [104] | | X | GNU GPL | 17/06/2009 | Windows, Linux, MAC OS X | Chercheurs et institutions diverses |
| NS-3 [113] | | X | GNU GPL | 20/08/2010 | Windows, Linux, MAC OS X | Chercheurs et institutions divers |
| OPNET [114] | X | | COPYRIGHT | 16/08/2010 | Microsoft, Red hat, Fedora | OPNET Technologies, Inc. Bethesda, Maryland, USA |
| OMNET++ [115] | | X | COPYRIGHT | 14/06/2010 | Windows, Linux, MAC OS X | Chercheurs et institutions diverses |
| GloMoSim [116] | | X | COPYRIGHT | 19/12/2001 | Redhat 7.2 | University de Californie, Los Angeles, USA |
| QualNET [117] | X | | COPYRIGHT | 01/02/2010 | Windows, Linux, MAC OS X | Scalable Network Technologies, Inc. Los Angeles, USA |
| NCTUns [118] | | X | COPYRIGHT | 15/01/2010 | Fedora 12 | Université de Chiao Tung, Taiwan |

Tableau 7.3 – Tableau comparatif des simulateurs de réseaux les plus connus

Au début de nos travaux, le simulateur NS-2 (*Network Simulator* version 2) était le simulateur de réseaux le plus réputé dans le domaine de la recherche en réseaux, venaient ensuite Opnet et GloMoSim loin derrière, et en dernier lieu Qualnet – une version payante plus complète de GloMoSim, Omnet++, NS-3 (*Network Simulator* version 3) et NCTUns [119]. Alors que la mise en œuvre d'une simulation nécessite d'écrire du code sur NS-2 ou sur NS-3, des simulateurs tels que OPNET, OMNET++, QualNET et NCTUns proposent une interface graphique à manipuler selon le principe du cliquer/déposer. Pour visualiser les scénarios, un animateur graphique peut être intégré au simulateur. Enfin, un simulateur peut comporter un outil d'analyse des résultats sous forme graphique.

Un aspect non moins important des simulateurs de réseaux est la question de passage à l'échelle. Certains d'entre eux sont, en effet, capables de s'exécuter en parallèle sur plusieurs ordinateurs comme GloMosim, QualNET et PDNS (Parallel/Distributed NS) qui est une version parallèle de NS-2. Quand on passe à l'échelle, les réseaux étudiés peuvent comporter des milliers de nœuds. Cela nécessite des temps de traitement considérables. D'où l'idée de répartir l'exécution sur plusieurs machines apportant chacune sa part de puissance de calcul.

#### 7.3.2.1 NS-3

NS-3 [113] est la génération suivante des simulateurs de type NS. Comme NS-2, il est gratuit et libre de droit sous licence GNU GPL. Son développement, initié en 2006, est assuré par une équipe de bénévoles et vise à dépasser les limites connues de NS-2 en prenant en compte les leçons tirées du passé. Cela couvre en particulier l'organisation générale, la modularité, les temps d'exécution des simulations et la consommation en mémoire. Dans cette perspective, alors que les modèles sont, comme dans NS-2, écrits en C++, l'utilisation du langage OTCL pour préciser les scénarios a été abandonnée. Les scénarios sont eux aussi décrits au moyen de C++. Par ailleurs, il est souvent possible d'appeler les objets C++ depuis les programmes écrits en Python, ce qui donne une grande souplesse de développement.

Alors que NS-2 implémente de nombreux modèles dont certains sont maintenant reconnus comme dépassés, l'écriture de NS-3 a été l'occasion de reprendre depuis NS-2 les modèles les plus utiles en même temps qu'en étaient développés de nouveaux conformes à l'état de l'art et aux besoins actuels de simulation [2]. Aujourd'hui, mais ce n'était pas le cas au début de nos travaux, NS-3 représente une solution mieux construite, plus solide, plus précise et plus efficace pour effectuer des simulations. Par ailleurs NS-2 était construit pour tenir compte des limitations, en puissance comme en capacité, des machines des années 1990, limitations que nous ne rencontrons plus aujourd'hui.

#### 7.3.2.2 OPNET

La société *OPNET Technologies incorporation* est propriétaire du logiciel de simulation de réseaux OPNET, plus connu aujourd'hui sous la désignation

2. Une partie des modèles développés dans NS-3 couvre plusieurs aspects de la norme 802.11, par exemple plusieurs modulations comme OFDM (*Orthogonal frequency-division multiplexing*) et DSSS (*Direct Sequence Spread Spectrum*), la technologie MIMO (*Multiple-Input and Multiple-Output*), la fonction ARF (*Automatic Rate Fallback*) de réduction automatique du débit, etc.

de *OPNET Modeler*. Ses laboratoires ont acquis une expertise dans le développement du simulateur et ses équipes se tiennent régulièrement informées des besoins des utilisateurs. Son chiffre d'affaire a augmenté de 40% au cours de la période 2004 2005.Grâce à ce logiciel, elle occupe une part de marché importante.

En ce qui concerne sa conception, OPNET, comporte globalement trois parties : une partie de construction de l'architecture à simuler, une deuxième de simulation et une troisième d'analyse de résultats. La construction est conçue de façon hiérarchique. Elle se fait par l'intermédiaire d'une interface graphique en plaçant les éléments du réseau sur une scène et en choisissant les protocoles de communication. Des outils de programmation sont aussi disponibles, par exemple pour définir les formats des paquets. La simulation est à événements discrets. Elle peut être accomplie par trois technologies différentes selon le type des résultats attendus.Enfin, des courbes, des diagrammes, des histogrammes et même à des animations font suite à l'analyse statistique des résultats.

Quelques détails techniques sont donnés dans le livre blanc d'OPNET. D'après ce document, le développement du logiciel est modulaire et orienté objet. En cas de simulations à grande échelle (e.g. scénarios sans-fil), il est possible de faire une exécution parallèle sur grille de calculs. Une bibliothèque de composants, avec leur code source, est aussi disponible. D'autres bibliothèques de composants externes peuvent aussi être intégrées à l'aide de l'interface graphique.

OPNET est surtout employé dans le monde industriel. C'est son prix, en grande partie, qui participe à son impopularité auprès des chercheurs universitaires. Une version complète d'OPNET coûte en effet près de 10.000$. Pour un doctorant, l'obtenir demande la constitution d'un dossier comportant des informations sur son sujet de thèse, son directeur, son laboratoire d'accueil, etc. Le prix est alors aux environs de 100$ pour une durée de licence de six mois. Il existe néanmoins une version gratuite de ce simulateur, c'est OMNET++.

#### 7.3.2.3 OMNET++

OMNET++ (*Objective Modular Network Testbed in C++*) est un simulateur à temps discret, gratuit et sous licence copyright, dont l'usage est connu aussi bien en recherche académique que dans l'industrie. Il est utilisé dans les domaines de réseaux de communications, de modélisation de logiciels complexes, de files d'attente, d'architectures matérielles, etc. D'une manière générale, il peut modéliser les systèmes où les entités communiquent par échange de messages et où l'approche à temps discret est applicable. Son développement a débuté en 1992 à l'université de Budapest. Aujourd'hui des chercheurs et diverses institutions contribuent à son développement et à sa maintenance. De plus des développeurs bénévoles apportent au code des contributions variées.

La principale caractéristique d'OMNET++ est sa modularité qui contribue en grande partie à la diversité de ses domaines d'application. Plusieurs *modules simples* sont en effet développés autour du noyau du simulateur. Des *modules composés* représentants les différents modèles sont ensuite obtenus par l'assemblage de plusieurs d'entre eux. C++ est le langage d'écriture des modules simples. La description d'un réseau s'effectue grâce au langage de haut-niveau de OMNET++, appelé NED (*Network Description*), qui permet de les assembler en modules composés. Une fois qu'une architecture est spécifiée, NED est transformé en langage C++ lors de la compilation.

Grâce à un ensemble de modèles assez complet développés ces dernières années, OMNET++ peut aujourd'hui constituer une bonne alternative à NS-2. De plus, il peut supporter des simulations à grande échelle comportant plus de 3000 nœuds. Son interface graphique, basée sur la plate-forme Eclipse, et son langage de haut niveau NED le rendent plus attractif que NS-2 ou NS-3 qui nécessitent l'écriture de code. Néanmoins, ses performances en terme de temps d'exécution sont en dessous de celles de NS-3. [120].

#### 7.3.2.4 NS-2

NS-2 (*Network Simulator* version 2) [104] est un simulateur à temps discret qui permet de simuler des protocoles du modèle TCP/IP, par exemple des protocoles de routage *unicast* comme AODV (cf. annexe A) ou *multicast* comme PUMA (*Protocol for Unified Multicasting Through Announcements*), des protocoles de transport divers comme TCP ou UDP, des protocoles de la couche application comme HTTP ou FTP et cela dans des environnements filaires ou sans fil utilisant les normes IEEE 802.11 et IEEE 802.15.4 ou communiquant avec des satellites.

Au départ, NS-2 a été développé comme une variante du simulateur *REAL network simulator* [121] en 1989 faisant ainsi partie des premiers efforts coordonnés de développement de simulateurs de réseau à temps discret. Depuis il n'a cessé d'évoluer. Son développement a été soutenu par la DARPA [17] en 1995 à travers le projet VINT (*Virtual InterNetwork Testbed*) [122] au LBL (*Lawrence Berkeley National Laboratory* à Berkeley, California) [123], Xerox PARC (*Palo Alto Research Center*) [3], UCB (*Université de Californie à Berkeley*) [124], et USC/ISI (*University of Southern California/Information Science Institute*) [125]. Entre 2001 et 2004, il a été soutenu par la DARPA à travers le projet SAMAN (*Simulation Augmented by Measurememt and Analysis for Networks*) [126] et par la NSF (*National Science Foundation*) à travers le projet CONSER (*Collaborative Simulation for Education and Research*) [127] en collaboration avec des chercheurs du ACIRI (*AT&T Center for Internet Research at ICSI, the International Computer Science Institute, Berkeley, California*) [128]. NS inclut aussi d'autres contributions de chercheurs ayant travaillé au projet Daedelus [129] de l'UCB, au projet Monarch (*Mobile Networking Architectures*) [130] de CMU (*Carnegie Mellon University*) [131] et à Sun Microsystems [4]. Depuis 2005, il est distribué sous licence GNU GPL (*General Public License*) [5]. Sa maintenance comme son développement sont assurés par des bénévoles qui le délaissent, peu à peu, au profit de NS-3, la génération suivant NS-2 (cf. section 7.3.2.1).

Le simulateur implante plusieurs modèles déterministes et stochastiques permettant de simuler non seulement des protocoles de réseaux filaires ou sans-fil, mais aussi l'architecture d'un nœud, la gestion de sa file, l'évolution de son énergie suite à l'activité en réseau, la propagation des ondes électromagnétiques, une topologie dynamique d'un réseau sans-fil, etc. Il comporte, par ailleurs plusieurs

3. Xerox PARC était un centre de recherche en informatique entre 1970 et 2002 situé à Palo Alto en Californie. Aujourd'hui, il s'appelle PARC et est une entreprise en recherche et développement dont Xerox reste le principal client mais l'entreprise a d'autres clients de co-développement.

4. Depuis 2010, Sun Microsystems a été racheté par Oracle

5. La Copyright de NS-2 peut être consultée sous le répertoire racine du code source de NS-2.

types d'ordonnanceurs [6] et un générateur de nombre pseudo-aléatoires [7]. Les scénarios construits peuvent être visualisés, à la fin des simulations, par le logiciel NAM (*NS Animator*).

Programmé au moyen de deux langage à objets (OTCL et C++), NS-2 est construit sur deux hiérarchies de classes [133] censées calquer fidèlement les éléments d'un réseau, les modules écrits en C++ implémentent naturellement les protocoles ainsi que les différents objets modélisés alors que les parties écrites en OTCL décrivent plutôt l'assemblage de ces divers éléments. Ainsi, dans un scénario de simulation, on instancie des objets de la hiérarchie OTCL et l'interpréteur se charge d'instancier leurs symétriques dans la hiérarchie C++. C++ sert à écrire les modèles et les algorithmes qui peuvent manipuler des fichiers de données assez volumineux mais qui, une fois finalisés, ne changent plus et sont compilés une fois pour toute. Cela répond au besoin d'accélérer l'exécution de ces parties complexes. OTCL sert à fixer les paramètres et les configurations des modèles et des algorithmes, ce qui ne prend qu'une part infime du temps d'exécution total mais permet d'éviter de re-compiler le simulateur à chaque nouveau scénario [8].

Nous constatons donc que la modification ou l'ajout d'un protocole nécessite la re-compilation de la totalité du code C++ du simulateur, ce qui peut constituer un frein à l'utilisation de NS-2 car celle-ci peut durer une bonne fraction d'heure. Plus généralement, l'installation de NS-2 est une tâche fastidieuse parce qu'elle nécessite, auparavant, le choix et l'installation de packages supplémentaires nécessaires au fonctionnement de NS-2. Néanmoins, ce simulateur présente l'avantage de pouvoir s'exécuter dans plusieurs environnements : Windows ce qui nécessite alors l'installation de l'émulateur UNIX Cygwin [9] [134], les systèmes de type Unix, les systèmes BSD et MAC OS X.

La mise en place d'une simulation nécessite l'écriture d'un fichier maître en OTCL qui instancie les objets, les configure en fixant les paramètres, prend en entrée le scénario de mouvement et le scénario de trafic s'il y en a, lance le simulateur et donne en sortie des fichiers de journalisation (en anglais *log files*) d'extension .tr qu'on appelle aussi fichiers de *traces* en plus de fichiers d'animation d'extension .nam exploitables par le visualisateur NAM. Les fichiers de traces peuvent aussi être complétés par d'autres fichiers *log* personnalisés en cas de besoin. Certains programmes d'analyse de traces sont proposés par des développeurs bénévoles et peuvent être téléchargés séparément. Mais il est, en général, nécessaire de les développer soi même pour répondre à des besoins spécifiques. Nous expliquerons plus en détails les outils et les données impliqués dans les simulations par NS-2 dans la section 7.4.1.

Le passage à l'échelle pour simuler des réseaux sans-fil, en particulier les réseaux ad hoc, peut impliquer des temps d'exécution importants pouvant aller jusqu'à plusieurs semaines. Des efforts de parallélisation ont été entrepris pour

6. Selon le manuel de NS-2, quatre types d'ordonnanceurs sont implémentés : *List scheduler*, *Heap scheduler*, *Calendar queue scheduler* et *Real-time scheduler*.

7. Le générateur de nombre seudo-aléatoire implémenté au sein de NS-2 est MRG32k3a (*combined Multiple Recursive Generator*) proposé par L'Ecuyer [132]. Ce générateur a été jugé assez bon pour que l'équipe NS-3 [113] en reprenne l'usage.

8. Dernièrement, certaines versions de NS-2, fournies par des développeurs indépendants, comportent un noyau compilé auquel peuvent être ajoutées des parties compilées séparément sans nécessiter la compilation de tout le code C++.

9. Cygwin est un logiciel pour Windows constitué d'un ensemble d'outils permettant d'avoir un environnement de travail POSIX.

développer PDNS [10] (*Parallel/Distribted NS*) [135] mais les efforts d'optimisation du temps d'exécution sont aujourd'hui concentrés dans la réalisation de NS-3.

#### 7.3.2.5 Notre choix

Répondre à la question « quel simulateur utiliser ? »n'est pas facile. De notre point de vue, NS-2 était un bon choix au début de cette thèse car :

- comme il est l'un des premiers simulateurs de réseaux à avoir été développés, et en raison de son caractère gratuit et libre de droit, la communauté scientifique a acquis une maturité quant à la validité et la performance de ses modèles contrairement aux simulateurs OPNET, OMNET++, QualNET et GloMoSim. Des retours d'expérience ont fait l'objet de plusieurs publications mettant en garde contre les pièges de simulations avec NS-2 [119], comparant les protocoles de routage pour les MANET [136], comparant les résultats de simulation à des résultats expérimentaux [137], etc. Ce phénomène rappelle le principe de *Kerckhoff* concernant la publicité des algorithmes cryptographiques (cf. section 2.4.1 du chapitre 2).
- NS-3 est un descendant de simulateurs de type NS et son développement a profité du retour d'expérience sur NS-2. Malheureusement, au début de cette thèse, en 2007, ni le port des modèles de NS-2 vers NS-3 ni la documentation de ce dernier n'étaient encore assez avancés. De plus la communauté scientifique y était moins expérimentée.
- la qualité de publications antérieures utilisant NS-2 donne des points de comparaison solides.

### 7.3.3 Modèles utiles et classification

Un simulateur comme NS-2 est composé de plusieurs petits modèles imbriqués. Chacun d'entre eux joue un rôle important dans la définition du plan de simulation et dans la qualité des résultats produits. C'est pourquoi il est important d'examiner cet aspect. Dans cette partie, nous les présentons selon les catégories suivantes : modèles de mouvement des nœuds, modèles de propagation et modèles de trafic. D'autres catégories existent comme celles des modèles de file d'attente, des modèles d'erreur et des modèles d'énergie. Mais nous nous concentrons sur les modèles qui nous semblent les plus intéressants dans cette thèse et les plus controversés.

Les modèles de mouvements font partie des modèles controversés dans le domaine de la simulation de réseau sans-fil. Nous en présentons quelques uns ci-dessous. Nous donnons aussi un aperçu de certains modèles de propagation.

#### 7.3.3.1 Modèles de mouvement

Une fois choisi le simulateur, la question suivante est celle de l'étude de mouvement des nœuds. Cette question avait été laissée de côté dans le chapitre précédent. Dans ce chapitre au contraire nous souhaitons que les nœuds se déplacent dans une aire fixée selon un mouvement qui donne une image fidèle des mouvements qui animent les nœuds d'un réseau réel. Plusieurs propositions ont

10. La dernière version est de 2004.

été faites en ce sens. Le choix que l'on peut faire entre elles dépend, en particulier, des propriétés théoriques des modèles proposés et des inférences qu'elle permet de faire. Elle dépend aussi de la qualité de la représentation intuitive qui correspond. Elle dépend enfin de la plus ou moins grande facilité qu'il y a à écrire les programmes de simulation correspondants.

L'examen de ces propriétés va tout d'abord être entrepris alors que l'on étudie le mouvement d'un seul nœud, $nd_0$ dont les coordonnées à l'instant $t$ sont $(x_0(t),\ y_0(t))$. Un premier modèle assez intuitif est le modèle *Brownien*. Dans ce modèle, les quantités $x_0(t)$ et $y_0(t)$ sont *à accroissements gaussiens indépendants* c.-à-d. que si $t_1 < t_2$ sont deux instants, $x_0(t_2) - x_0(t_1)$ suit une loi $\mathcal{N}\ (0,\sigma\sqrt{t_2 - t_1})$, la même relation existant pour $y_0(t_2)$ et $y_0(t_1)$.

Par ailleurs, si $t_1 < t_2 < t_3 < t_4$, alors $x_0(t_2) - x_0(t_1)$ est indépendante de $x_0(t_4) - x_0(t_3)$. C'est à dire que les coordonnées $x_0$ et $y_0$ suivent au long du temps des *mouvements browniens* que l'on supposera d'ailleurs indépendants.

Les propriétés du *mouvement brownien* sont bien connues et ont fait l'objet d'études approfondies [138].

Pourtant, ce modèle souffre de plusieurs défauts : si les trajectoires sont bien continues, elles ne sont nulle part dérivables, ce qui rend ce mouvement, *sans vitesse*, incompatible avec l'idée que l'on se fait du déplacement d'un nœud mobile. L'additionnalité des variances montre que l'écart type de $x_0(t)$ est $\sigma\sqrt{t}$ ce qui montre que le nœud n'est pas confiné dans un espace fini. Les *améliorations* que l'on peut faire à ce modèle en faisant par exemple réfléchir la trajectoire du nœud sur les côtés du domaine de référence amènent à des calculs compliqués et à la perte des qualités de simplicité de ce modèle. Pour ces raisons, il n'est guère utilisé.

Un autre modèle est le modèle de direction aléatoire (en anglais *Random direction*) qui considère les mouvements individuels des nœuds en l'absence de tout obstacle. Il a été créé pour maintenir une distribution spatiale uniforme des nœuds dans une région délimitée. Un nœud choisit selon une loi uniforme une direction dans l'intervalle $[0, 2\pi]$ et, selon une loi uniforme, une vitesse dans l'intervalle $[v_{min}, v_{max}]$. Il commence ensuite à se déplacer uniformément vers cette direction à la vitesse choisie. Lorsqu'il arrive aux bords de la région, il choisit, de la même manière, une nouvelle vitesse et une nouvelle direction vers laquelle il se dirige à cette vitesse. Il réitère ces étapes jusqu'à la fin de la durée de la simulation.

L'un des modèles de mouvement aléatoire le plus utilisé dans la simulation de réseaux MANETs, en particulier celles menées avec NS-2, est le modèle stochastique RWP (*Random Way-Point*). Ses principales caractéristiques sont les suivantes :

1. la durée totale de la simulation, notée $T$, les vitesses minimale et maximale, notées respectivement $v_{min}$ et $v_{max}$, le temps moyen de pause [11], noté $t_p$, sont fixés à l'avance.
2. le nombre de nœuds dans le réseau est connu à l'avance et ne change pas au cours du temps. On le note $N$ et on note les nœuds $nd_i$ pour $i \in \{0, ..., N-1\}$
3. les nœuds sont placés et se déplacent sur une surface plane $S$ de dimensions $X$ et $Y$.

11. Le temps de pause est un temps d'arrêt marqué par un nœud entre deux déplacements successifs.

4. les positions initiales des nœuds sont tirées au hasard. Soient ($x_i$, $y_i$) les coordonnées du nœud $nd_i$ à $t = 0$ s, alors la valeur de $x_i$ est tirée selon une variable aléatoire uniforme entre X et Y. Idem pour $y_i$.
5. pour chaque nœud $nd_i$, une date de début de mouvement $t_{0_i}$ est tirée suivant une variable aléatoire uniforme dans $[0, T]$, une vitesse $v_i$ est tirée selon une variable aléatoire dans $[v_{min}, v_{max}]$ et une position $P'_i$ =($x'_i$, $y'_i$) dont les coordonnées sont tirées respectivement selon une variable aléatoire uniforme dans $[X, Y]$. A $t = t_{0_i}$, $nd_i$ se dirige vers $P'_i$ avec un mouvement rectiligne uniforme de vitesse $v_i$. Quand il arrive à $P'_i$, il s'arrête pendant la durée $t_p$.Quand cette durée est écoulée, une autre vitesse $v_i$ et une autre position $P'_i$ sont tirées au hasard et $nd_i$ se dirige alors vers cette nouvelle position à la nouvelle vitesse. Ce calcul est réitéré jusqu'à écoulement de la durée de simulation.

La question qui se pose avec ce modèle est celle de l'ergodicité de la distribution des nœuds. C'est elle qui garantit que des moyennes effectuées au cours du temps correspondent bien à des estimateurs des espérances. Une garantie de cette ergodicité serait, d'après le théorème de Birkhoff-Doob [139], que le processus soit stationnaire au second ordre. Or de nombreuses critiques ont été faites montrant que cette stationnarité n'a pas lieu dans le modèle tel que décrit ci-dessus.

En particulier certaines d'entre elles ont montré que la distribution globale des nœuds dans l'espace, à long terme, se concentre au milieu de la région de déploiement [140]. D'autres ont montré que la vitesse moyenne des nœuds à un instant donné, lorsque $v_{min}$ est prise égale à 0, diminue au cours du temps de la simulation [141]. Une solution approchée, mais non rigoureuse, consiste alors à ne pas prendre en compte les résultats fournis par une simulation pendant un certain intervalle de temps initial suffisamment long et de ne compter, par la suite, que les valeurs calculées, à l'état de stabilité supposée du système. Néanmoins, la longueur de cette intervalle ne semble pas être facile à déterminer et certains travaux se sont avérés invalides pour ces raisons [119].

Puisque l'on a vu qu'une distribution initiale uniforme des points amenait à un processus non stationnaire, il est plus productif de distribuer les points selon une distribution initiale stationnaire. Dès lors, le théorème de Birkhoff-Doob s'applique et l'on a un processus ergodique. Naturellement cette distribution stationnaire n'est plus uniforme. Elle s'en approche toutefois suffisamment pour donner des résultats de qualité. Pour obtenir un nuage de points répondant à ces contraintes, on peut utiliser la version *stationnaire* de RWP, distribuée grâce au *package mobgen-ss*, développée par l'équipe Toilers au sein de l'école des Mines du Colorado (*Colorado School of Mines*) [142]. C'est la solution que nous avons retenue.

En pratique, nous avons utilisé plusieurs dizaines de nœuds ayant chacun une trajectoire, tirée au hasard de façon indépendante, conforme au même modèle.

#### 7.3.3.2 Modèles de propagation

Un modèle de propagation décrit l'affaiblissement de la puissance d'un signal émis par un nœud émetteur $e$ lors de son parcours vers un nœud récepteur $r$. Cela permet de définir une zone de réception du signal de $e$ où les nœuds se trouvent en lien directe avec $e$ et, partant, de trouver la topologie du réseau en fonction des situations des nœuds, c.-à-d. de sa géométrie.

Dans le modèle déterministe modélisant la propagation en espace libre (en anglais *free space*), on suppose qu'une ligne d'horizon existe toujours entre l'émetteur et le récepteur c.-à-d. que le chemin de propagation ne comporte pas d'obstacles. La puissance à la réception est alors proportionnelle à l'inverse du carré de la distance $d$ entre $e$ et $r$. La zone de réception est un disque de centre $e$ et de rayon proportionnel à la racine carrée de la puissance d'émission.

Le modèle *free space* n'étant pas très fidèle à la réalité, pour mieux s'en approcher, le modèle de propagation à deux rayons avec réflexion par le sol (en anglais *two-ray ground reflection*), qui est aussi déterministe, prend en compte, en plus du chemin de propagation directe entre l'émetteur et le récepteur, le chemin de propagation du signal réfléchi par le sol. La puissance à la réception est alors proportionnelle à l'inverse de $d^4$ et aux hauteurs de $e$ et de $r$. La zone de réception est un disque de rayon proportionnel à la racine quatrième de la puissance d'émission. Le modèle *shadowing* généralise les deux modèles précédents.

Une étude menée par W. Xiuchao à l'université nationale de Singapour [143] préconise d'utiliser le modèle *two-ray ground reflection* quand il s'agit de simulation de propagation dans un milieu à l'extérieur des bâtiments en présence ou en absence de mouvement et le modèle *shadowing* lorsqu'il s'agit de milieu intérieur aux bâtiments.

#### 7.3.3.3 Modèles de trafic

Le modèle de trafic définit les dates d'envoi des paquets par un nœud du réseau ou par l'ensemble des nœuds. Il peut modéliser en plus la taille des données transmises à chaque envoi. NS-2 offre quatre sortes de profils : trafic exponentiel, trafic Pareto, trafic CBR (*Constant Bit Rate*) et processus poissonien.

Un trafic exponentiel comporte des périodes d'activité où des paquets de tailles identiques sont délivrés à débit constant et des périodes d'absence d'activité où aucun paquet n'est envoyé. Les intervalles d'activité ainsi que les intervalles d'absence d'activité sont tirés au sort selon deux lois exponentielles de moyennes différentes. Le profil d'un trafic Pareto est semblable à celui d'un trafic exponentiel à la différence des intervalles d'activité et d'absence d'activité qui sont tirés selon une loi stable de Pareto qui sert plutôt dans l'étude des réseaux de données à grande vitesse. Un trafic de type CBR est formé de paquets de taille constante envoyés à un débit déterminé. De l'aléa peut être introduit dans les intervalles séparant deux paquets successifs. Enfin, un processus poissonien modélise les événements rares comme les arrivées dans un intervalle de temps s'il est suffisamment petit. Les longueurs des intervalles entre deux arrivées successives sont tirées selon une loi exponentielle de même paramètre que la loi de Poisson.

#### 7.3.3.4 Nos choix

Cette partie récapitule les choix de modèles que nous avons faits. Ces choix ne sont pas restreints aux classes présentées ci-dessus mais s'étendent à d'autres classes de protocoles suivant les couches de réseau.

| Modèle | Nom | Paramètres |
|---|---|---|
| mouvement | RWP stationnaire | vitesse, pause, superficie, nombre de nœuds, durée de simulation |
| propagation | *two-ray ground reflection* | puissance d'émission, seuils de réception |
| trafic | exponentielle | |
| | CBR | débit |
| couche de liaison de données | 802.11 | modulation, temporisateurs |
| protocole de routage | AODV | (paramètres par défauts) |
| protocole de transport | (cf. section 7.4.2) | |
| protocole applicatif | (cf. section 7.4.2) | |

Tableau 7.4 – Choix de modèles, protocoles et paramètres associés

## 7.4 Travail préparatoire aux simulations dans NS-2

### 7.4.1 Organisation des outils et des données

En plus du logiciel visualisateur NAM qui complète NS-2, d'autres petits programmes peuvent venir le parfaire, par exemple des générateurs de mouvement ou de trafic, et aussi des analyseurs de résultats de simulations. Un logiciel de tracé peut aussi être employé pour tracer des graphes mettant en lumière les résultats analysés.

La figure 7.2 illustre l'ensemble de ces outils dans des rectangles et le flux des données qui circulent entre eux par des ovales. Certains exemples d'outils sont précisés en italique. Ce sont ceux que nous avons utilisés. Lorsqu'il n'y a pas d'exemples, il s'agit d'outils que nous avons développés. Ainsi, le générateurs de mouvement est *mobgen-ss*, la version *stationnaire* de RWP, et le logiciel de tracés est Gnuplot. Le générateur de trafic est un ensemble de scripts OTCL que nous avons écrits. Les analyseurs de traces sont des programmes en C ou des scripts SHELL que nous avons produits selon les données que nous souhaitions tracer.

Les données sont de deux types : les données d'entrée et les données de sortie. Les données d'entrée sont l'ensemble des paramètres à fixer pour les modèles de mouvement, de trafic et autres modèles sélectionnés pour la simulation. Un script OTCL maître centralise la description des modèles et des paramètres associés. Il les donne, ensuite, à NS-2. Un ensemble de scripts SHELL maîtres a été développés pour automatiser la paramétrisation et le lancement des outils par la suite.

Les données de sortie sont de plusieurs types. Celles notées dans les fichiers de mouvement et de trafic sont sous formes de script OTCL et sont données par le script maître au simulateur. Celles produites par NS-2 sous forme de

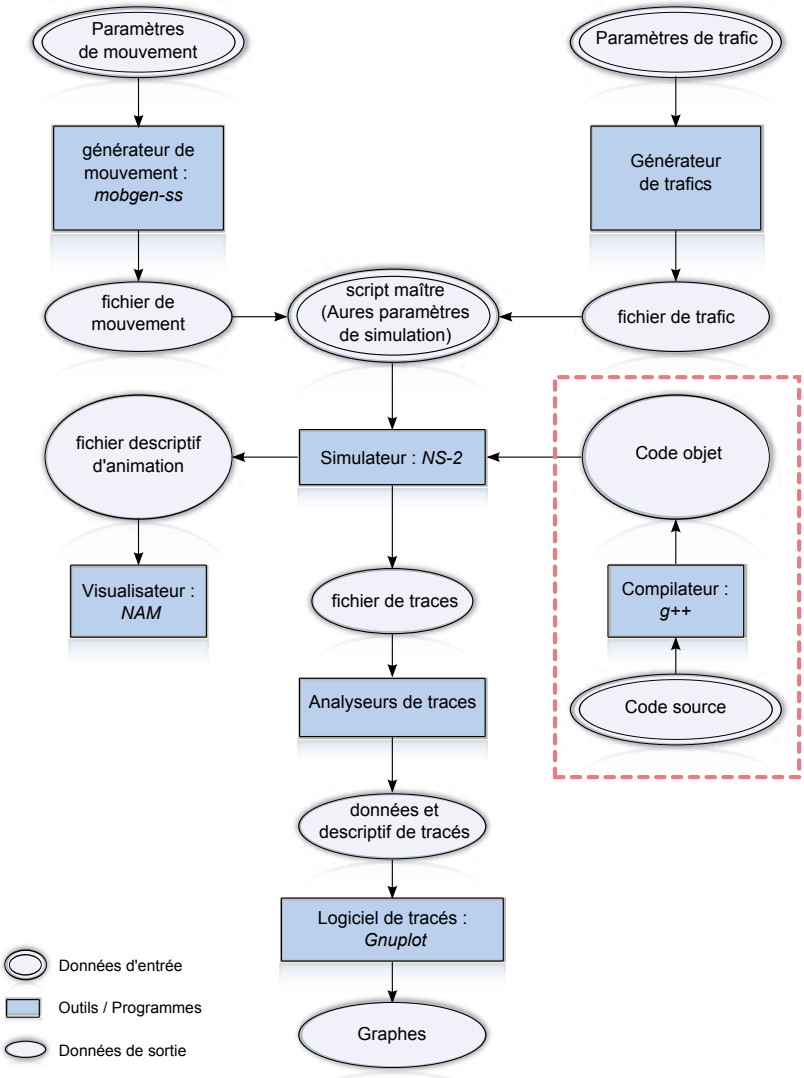


FIGURE 7.2 – Diagramme de flux

traces des événements sont traitées par les analyseurs et sous forme de descriptif d'animation sont traitées par NAM. Enfin, celles générées par les analyseurs sont tracées par Gnuplot et visualisées en graphes.

Le morceau du schéma entouré par un cadre en pointillés suit aussi le même principe d'entrée puis traitement par un outil puis sortie des données. En revanche, il ne fait pas partie réellement de la préparation de simulation mais de l'*extension* du simulateur. Ainsi, un code C++, implémentant un protocole au sein de NS-2 et compilé avec le code source de celui-ci, produit un code objet qui s'exécute à l'appel du simulateur. L'implémentation du protocole et la compilation qui s'ensuit se font en amont avant le lancement des simulations [12].

12. Ainsi que cela a été expliqué à la section 7.3.2.4, on sépare la tâche de compilation, longue en temps d'exécution, de la tâche de simulation proprement dite.

La section suivante décrit l'extension de NS-2, dans sa version 34 (NS-2.34), par l'implémentation du protocole d'authentification.

### 7.4.2 Extension de NS-2 par une implémentation du protocole d'authentification

La difficulté essentielle dans la simulation de notre protocole d'authentification réside dans le fait qu'un JN doit attendre un certain nombre de réponses avant de lancer les messages de la deuxième phase. Ce comportement est, à notre connaissance, inexistant parmi les protocoles implémentés dans les simulateurs de réseaux. Pour cette raison, une première aventure d'implémentation avec OTCL en simulant les messages d'authentification par des messages UDP était difficile à mener jusqu'au bout.

Nous avons donc opté pour une implémentation en C++ au cœur de NS-2. Ce choix s'est avéré salutaire notamment en raison de la richesse de la bibliothèque de classes lorsqu'il a fallu manipuler des temporisateurs et lorsqu'il a fallu distinguer les traitements des données d'authentification d'autres types de traitements. De plus, dans les fichiers de traces, il a ainsi été possible de marquer les messages d'authentification.

#### 7.4.2.1 Cahier des charges et simplification

Il s'agit de développer le protocole d'authentification au niveau application. Selon le modèle en couches de protocoles (cf. section 2.2.3 du chapitre 2), les messages de cette application doivent pouvoir être traités par la couche de transport. A des fins de simplification et d'efficacité, nous avons choisi de la baser sur le protocole UDP. En effet, il nécessite moins de signalisation que TCP, qualité appréciable dans les MANETs. De plus, les cas de défaillances du protocole d'authentification sont déjà pris en charge grâce à l'algorithme à multiples tentatives, donc la possibilité de retransmission de paquets offerte par TCP n'est pas forcément utile.

D'autre part, comme le traitement des messages d'authentification prend un temps supposé négligeable (cf. tableau 6.2 et section 6.4.1 du chapitre 6), la couche application peut faire abstraction du détail du contenu de ces messages. Il lui suffit alors de connaître la taille des messages et l'ordre dans lequel ils doivent être envoyés. Elle doit aussi être capable de construire l'en-tête des messages qu'elle émettra à la couche de transport et de les traiter lorsqu'elle en recevra en provenance celle-ci . Enfin, elle doit manipuler des temporisateurs pour le lancement de nouvelles tentatives ou le renouvellement de l'authentification en cas d'expiration du jeton d'accès.

La couche de transport doit pouvoir distinguer un message d'authentification qui arrive ou qui doit être émis de tout autre type de message. Elle doit donc savoir analyser l'en-tête d'un message d'authentification. Elle doit, par ailleurs, segmenter et reconstituer un message en fonction de sa taille.

Enfin, la différence de comportements entre un client et un pair, au sens Dist-AAA (cf. sections 4.3 et 4 du chapitre 4.4), doit être considérée.

#### 7.4.2.2 Création de nouvelles classes

Développer une telle application n'est pas une tâche simple car cela nécessite d'intervenir à divers niveaux de la hiérarchie de classes [133] de NS-2. Certaines directives peuvent, d'ailleurs, être consultées sur [144].

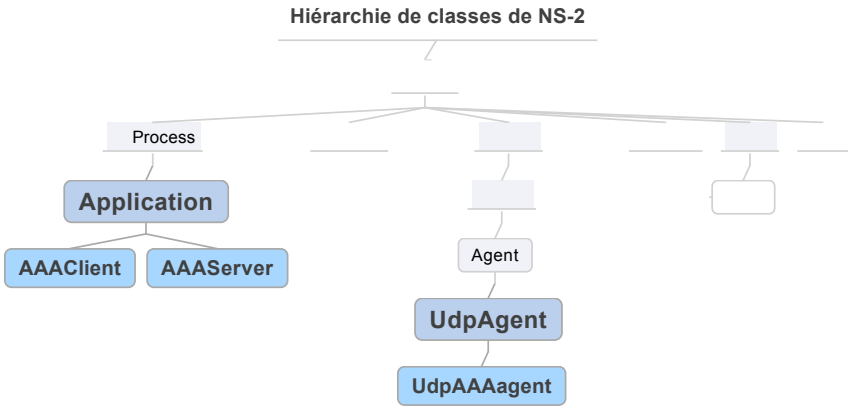


FIGURE 7.3 – Nouvelles sous-hiérarchies dans la hiérarchie de classe de NS-2

Nous appelons cette application *AAAClient* au niveau d'un client et *AAAServer* au niveau d'un pair AAA. Après examen de la hiérarchie de classes, nous avons décidé d'implémenter *AAAClient* et *AAAServer* comme deux classes dérivées de la classe *Application*. Afin que la couche transport soit compatible avec cette nouvelle application, nous avons dérivé une classe appelée *UdpAAAagent* de la classe *UdpAgent*. Deux nouvelles sous-hiérarchies ont été donc créées comme illustré, à titre indicatif, par la figure 7.3. On voit qu'elles ne sont pas issues des mêmes parents.

D'autres classes supplémentaires ont été définies pour refléter ces nouvelles sous-hiérarchies, le nouvel en-tête et les temporisateurs dans la hiérarchie symétrique OTCL.

Afin que ces modifications de l'architecture de NS-2 soient prises en compte lors de l'installation, les fichiers objets relatifs à ce module ont été déclarés dans le fichier *Makefile.in*.

#### 7.4.2.3 En-tête AAA et numéro de séquence

Un en-tête AAA, appelé $hdr_{aaa}$, a été défini dans la classe *UdpAAAagent* qui est aussi manipulé par les classes *AAAClient AAAServer*. Il s'agit d'une structure contenant le numéro de séquence du message AAA d'authentification envoyé, sa taille en octets et la date de son émission ou de sa réception. Afin qu'il soit connu dans NS-2, il a ajouté dans la pile de formats d'en-têtes définie dans le fichier *"ns-packet.tcl"*.

Lorsqu'un client rejoint le réseau MANET, il ne s'est pas encore authentifié. Ses premiers messages d'authentification auront comme numéro de séquence 0. A chaque nouvelle étape, ce numéro est incrémenté de un. La valeur du numéro de séquence d'un message d'authentification se trouve alors définie ainsi :

- 0 *mod* 4 : si le message est envoyé par un JN à un pair AAA lors de la première étape d'une tentative d'authentification,
- 1 *mod* 4 : si le message est envoyé par un pair AAA lors de la deuxième étape de cette tentative en réponse à la demande du JN,
- 2 *mod* 4 : si le message est envoyé par le JN au pair AAA lors de la troisième étape de la tentative,
- 3 *mod* 4 : si le message est envoyé par le pair AAA lors de la quatrième étape de la tentative en réponse au message du JN émis à la troisième étape.

De plus, tous les messages envoyés lors d'une même étape, que ce soit par le JN ou par les pairs AAA, ont le même numéro de séquence.

L'incrémentation en continu des numéros de séquence sans retour à 0 permet de distinguer, plus tard, dans les fichiers de traces, les différentes tentatives d'authentification pour chaque client.

#### 7.4.2.4 Classe *UdpAAAagent*

Cette classe étend la classe *UdpAgent* pour le support de l'application AAA. Pour cela, nous avons ajouté à sa classe mère, *Agent* (cf. FIGURE 7.3), les prototypes des méthodes : *supportAAA()* et *enableAAA()* définies dans *UdpAAAagent*. Ainsi, les fonctions suivantes sont devenues opérationnelles :

- envoi des données reçues de l'application AAA, que ce soit de la classe *AAAClient* ou de la classe *AAAServer*, après avoir mis en forme l'en-tête AAA et ajouté les ports et les adresses IP de la source et de la destination.
- réception des données destinées à l'application AAA.
- segmentation ou assemblage des données si nécessaire.

La spécification des ports et des adresses IP permet à la classe *AAAClient* d'identifier les différents pairs AAA puisqu'une étape du protocole d'authentification implique plusieurs d'entre eux et non un seul.

#### 7.4.2.5 Classe *AAAClient*

A travers le script maître de la figure 7.2 de la section 7.4.1, la classe du client prend en entrée les valeurs des variables : $n$, $th_M$, $th_i$, $th$, $K$, $t_{max}$ et T, la durée d'expiration d'un jeton (cf. tableau 7.5). Ces dernières doivent avoir été définies, auparavant, avec leurs valeurs par défaut, dans le fichier *ns-default.tcl*.

*AAAClient* définit un certain nombre de méthodes permettant d'initialiser le client lorsqu'il arrive dans le réseau, de l'arrêter lorsqu'il part, de lancer une authentification, d'envoyer un message ou un ensemble de messages destinés à plusieurs pairs, de recevoir un message, de planifier la prochaine tentative d'authentification en cas d'échec et le prochain processus d'authentification à l'expiration du jeton, de compter le nombre de messages arrivés au cours d'une étape, d'initialiser ce nombre lorsque cette étape est terminée, de choisir les pairs à contacter, etc. Deux possibilités ont été prévues pour le choix des pairs à contacter lors de la première étape d'une authentification à un instant donné :

1. $th_M$ pairs les plus proches du JN
2. $th_M$ sélectionné aléatoirement parmi les $n$ pairs

A la troisième étapes, les pairs contactés sont les $th_i$ qui ont été les plus réactifs c.-à-d. ceux qui ont répondu les premiers.

| Modèle | Nom | Paramètres |
|---|---|---|
| protocole de transport | UDP | |
| protocole applicatif | AAA d'authentification | $n$, $th_M$, $th_i$, $th$, $K$, $t_{max}$, T, pairs AAA choisis parmi les plus proches, pairs choisis aléatoirement |

TABLEAU 7.5 – Protocoles et paramètres associés (suite)

**Envoi de messages** Une trace de l'événement de lancement d'une nouvelle tentative est enregistrée dans un fichier de traces particulier, appelé *authTraces.tr*, que nous avons spécifié pour les seuls événements se rapportant aux authentifications des clients. Les informations enregistrées incluent l'identifiant du client et la date de lancement récupérée par l'appel de l'ordonnanceur.

L'envoi des messages utilise deux types de temporisateurs :

- *timeout* : un temporisateur d'expiration de la durée maximale d'attente $t_{max}$, définie dans l'algorithme à tentatives multiples. Avant d'être programmé, la classe *AAAClient* vérifie que le nombre de tentatives maximal $K$ n'a pas été atteint.
- *new_ auth* : un temporisateur d'expiration de la durée de validité du jeton.

**Réception de messages** Le client n'effectue pas les même opérations selon qu'il est à la deuxième ou à la quatrième étape :

- à la 2e : les messages réponses des pairs sont reçus un à un de la classe *UdpAAAagent*. Ils sont comptés et les identifiants des pairs, numérotés de 0 à $n-1$, sont au fur et à mesure enregistrés dans un tableau. Ainsi les pairs se trouvent classés par ordre de réactivité. Lorsque leur nombre atteint $th_i$, *AAAClient* lance la 3e étape.
- à la 4e : de la même manière, les réponses sont comptées. Lorsque leur nombre atteint $th$, la tentative est considérée comme réussie et l'événement est enregistré dans le fichier *authTraces.tr*.

**Numéro de séquence interne** Un client tient un numéro de séquence, *seq_*, interne afin de savoir à tout moment où il en est dans le processus d'authentification. C'est la valeur de ce numéro qu'il copie dans ses messages.

La figure 7.4 montre l'évolution de la valeur de *seq_* en fonction des événements :

- au départ, quand le client arrive, son numéro de séquence *seq_* est égal à 0. C'est la valeur qu'il copie dans les messages envoyés à la première étape.
- une fois ces messages émis, *seq_* est incrémenté. Plus tard, lorsque les réponses des pairs arriveront, la valeur des numéros de séquence qu'elles contiennent est comparée à celle de *seq_*.
- *seq_* garde la valeur 1 tant que le nombre de réponses n'a pas atteint $th_i$. Si le temporisateur *timeout* expire, la valeur de *seq_* revient à 0 et la tentative est abandonnée. Toute réponse ultérieure est supprimée silencieusement. Un enregistrement dans le fichier *authTraces.tr* note l'identifiant

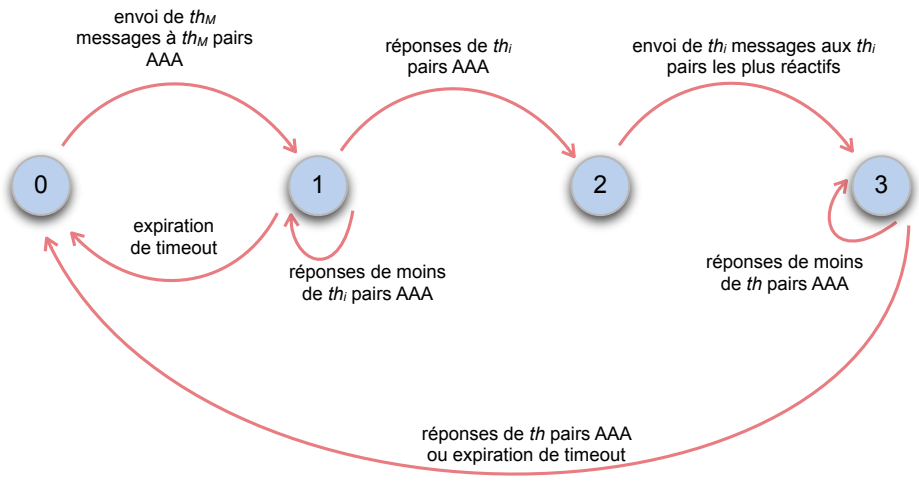


FIGURE 7.4 – Diagramme d'état du numéro de séquence interne de *AAAClient*

du client, la date à laquelle il a abandonné la tentative, l'étape à laquelle la tentative a été abandonnée et le nombre de réponses qui ont été reçues jusque là.

- si le nombre de réponses atteint $th_i$, $seq_$ est incrémentée ($seq_$=2). Le client lance la 3e étape en émettant $th_i$ messages vers les pairs qui étaient les plus rapides à répondre.
- une fois ces messages émis, $seq_$ est incrémenté à nouveau ($seq_$=3). Plus tard, lorsque les réponses des pairs arriveront, la valeur des numéros de séquence qu'elles contiennent est comparée à celle de $seq_$.
- $seq_$ garde la valeur 3 tant que le nombre de réponses n'a pas atteint $th$. Si le temporisateur *timeout* expire, la valeur de $seq_$ revient à 0 et la tentative est abandonnée. Toute réponse ultérieure est supprimée silencieusement. Un enregistrement dans le fichier *authTraces.tr* note l'identifiant du client, la date à laquelle il a abandonné la tentative, l'étape à laquelle la tentative a été abandonnée et le nombre de réponses qui ont été reçues jusque là.
- si le nombre de réponses atteint $th_i$, $seq_$ est incrémentée. Sa valeur modulo 4 vaut donc 0. C'est pourquoi une flèche allant vers 0 a été dessinée sur la figure 7.4. L'événement de réussite de la tentative est enregistré dans le fichier *authTraces.tr* comme décrit ci-dessus. Le client ne lancera plus de nouvelle tentative. Il est, en revanche, prêt à lancer un nouveau processus d'authentification dès que le temporisateur aura expiré *new_auth*.

Il est à remarquer qu'ainsi définies, les valeurs prises par $seq_$ ne cessent en réalité de croître. Les valeurs affichées dans les cercles sur la figure 7.4 sont en fait les valeurs de $seq_$ modulo 4. Ce choix a été opéré afin de pouvoir plus tard distinguer les différentes tentatives dans le fichier *authTraces.tr* grâce au numéro de séquence.

#### 7.4.2.6 Classe *AAAServer*

Comme la classe *AAAClient*, la classe *AAAServer* inclut les fonctions d'initialisation, de lancement, d'arrêt, d'envoi et de réception de messages. Mais contrairement à celle-ci, elle n'implémente pas la gestion de l'envoi ou de la réception de plusieurs messages à la fois. Quant au numéro de séquence interne, elle n'en définit pas non plus car un pair AAA n'a pas besoin de savoir à tout moment dans quelle étape il est. Cela est d'autant plus vrai qu'il peut être, en plus, en train d'authentifier et d'autoriser plusieurs clients à la fois. Ainsi, le calcul du numéro de séquence à ajouter dans l'en-tête d'une réponse se calcule simplement en incrémentant celui du message requête qui l'a généré. Un soin reste à observer quant à la longueur de la réponse selon que le numéro de séquence du message reçu est égal à 0 $mod$ 4 ou à 2 $mod$ 4.

#### 7.4.2.7 Traces AAA

En plus des événements de début, de fin et d'abandon d'authentification enregistrés dans le fichier *authTraces.tr*, nous avons défini d'autres types de traces. Il s'agit d'enregistrer plus généralement les événements d'envoi, de réception et de suppression de messages AAA d'authentification dans le fichier de traces classique de NS-2. C'est le fichier dans lequel sont journalisés habituellement tous les événements qui ont eu lieu pendant la simulation. Dans ce but, nous avons modifié les fichiers *cmu-trace.cc* et *trace.cc* pour qu'à chaque fois qu'un message AAA d'authentification passe, NS-2 le note.

## 7.5 Scénarios de simulations et résultats

Nous avons exploré plusieurs scénarios de simulations utilisant NS-2.34 (cf. section 7.3.2) et les modèles récapitulés à la section 7.3.3.4. Le but étant d'évaluer les conditions et les paramétrages pour lesquels les performances du protocole AAA d'authentification sont optimales. A travers le choix des valeurs des seuils intermédiaires $th_M$ et $th_i$, la section 7.2 a montré qu'il existe un lien étroit entre le taux de réussite et la conception même du protocole. D'autres choix et paramètres peuvent aussi avoir un impact.

Nous proposons de fixer de manière définitive les paramètres des modèles de mouvement et de propagation ainsi que les paramètres de la couche physique et de la couche liaison de données, sachant que le protocole de routage est AODV (cf. annexe A). De plus, afin que de se placer dans un réseau dont le trafic est proche d'un réseau normal, nous introduisons *un trafic ambiant* en plus du trafic AAA d'authentification. Une fois cela fait, l'étude du protocole nécessite de faire un choix dans chacune des catégories suivantes :

1. modèle du trafic d'authentification
2. manière dont les pairs sont choisis par un nœud arrivant avant de lancer l'authentification
3. valeurs des trois seuils ($th_M$, $th_i$, $th$)

La section suivante précise les différents paramétrages que nous fixons. Elle donne, par ailleurs, les choix possibles dans chacune des catégories précédentes.

### 7.5.1 Paramétrages généraux

#### 7.5.1.1 Mouvement

Nous avons considéré un réseau MANET composé de 100 nœuds qui se déplacent sur une surface de superficie 600mX600m à une vitesse maximale de 10m/s et un temps de pause de 0 s (cf. tableau 7.6). Cela peut être rapproché par

| Paramètre | Valeur |
|---|---|
| nbr de nœuds | 100 |
| nbr de pairs AAA | 20 |
| nbr de JNs | 80 |
| superficie | 600mX600m |
| vitesse des nœuds | moyenne=5.2m/s, delta=4.75m/s, max=10m/s |
| temps de pause | moyenne=0 s , delta= 0 s |
| durée de simulation | 4000 s soit 01 :06 :40 |
| carte sans fil | ORiNOCO 802.11b |

Tableau 7.6 – Résumé des paramétrages

un réseau fonctionnant dans un bourg comme celui d'Ars en Ré ou d'un quartier d'une grande ville comme celui de la montagne Sainte-Genevève à Paris (5e). 20% des nœuds c.-à-d. 20 nœuds sont des pairs AAA, ce qui est largement suffisant dans un tel réseau. Les 80 nœuds restants sont des nœuds ordinaires qui doivent être authentifiés et autorisés avant de pouvoir envoyer leur trafic sur le réseau (cf. Figure7.5). Pour ce qui est du modèle de mouvement, les arguments évoqués à la section 7.3.3.1 nous ont amené à utiliser RWP en mode stationnaire grâce au package *mobgen-ss* [142]. Ce package nous a permis de définir une distribution stationnaire à la fois des positions et des vitesses des nœuds.

#### 7.5.1.2 Couche sans-fil et propagation

Pour la transmission sans-fil, on a paramétré les modules Phy/WirelessPhy et Mac/802_11 de NS pour simuler une carte WIFI de type ORiNOCO[13] 802.11b comme indiqué dans [143] [145]. Il n'y a pas d'erreurs de canal. Le modèle de propagation utilisé est le modèle *two-ray ground reflection* (cf. tableau 7.6).

#### 7.5.1.3 Trafic ambiant

Afin d'être le plus proche possible d'un cas réel, les simulations des authentifications ont été réalisées dans ce que nous considérons comme un *réseau*

13. ORiNOCO (en français Orénoque) est le nom d'une carte WiFi très répandue commercialisée par la firme Proxim Wireless Corp. www.proxim.com/cat-orinoco-client-products.cfm. Elle a été souvent utilisée comme référence dans des travaux antérieurs ce qui permet de garder à l'esprit des ordres de grandeurs commodes [143].

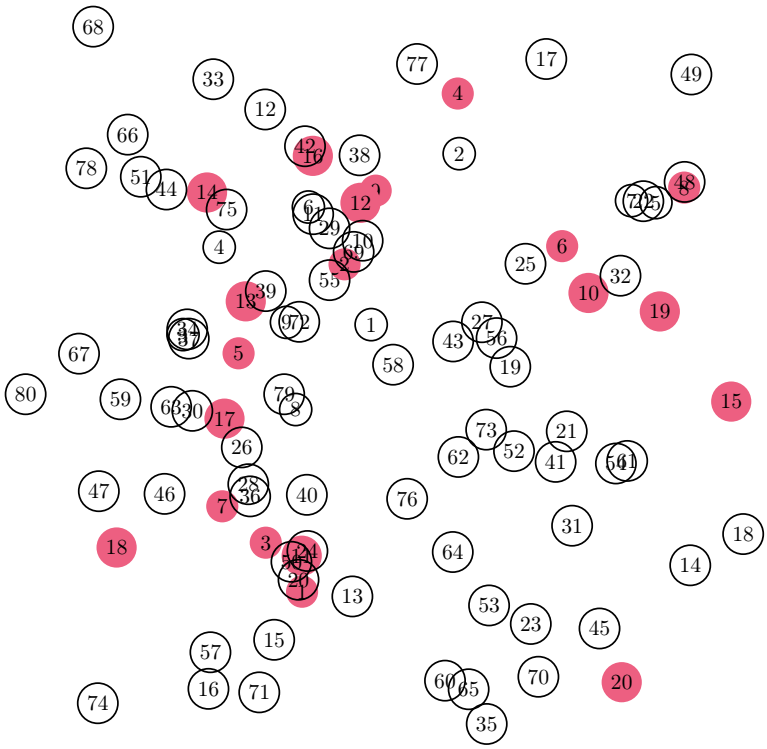

FIGURE 7.5 – Positions initiales des clients et des pairs

*modérément chargé en trafic* opérant avec une *vitesse de croisière*. Cela consiste à injecter du trafic supplémentaire, en plus du trafic d'authentification, simulant d'autres types de trafic dans un réseau, par exemple du trafic de données ou de signalisation. La simulation de ce trafic est réalisée par du trafic Constant Bit Rate (CBR) [14] dont le débit a été déterminé par des simulations préalables et en appliquant la méthode de dichotomie comme expliqué ci-dessus.

Comme tous les nœuds jouent le même rôle en ce qui concerne le trafic CBR et puisqu'ils sont à la fois positionnés et se déplacent de façon aléatoire, il est inutile de choisir aléatoirement les sources et les destinataires du trafic CBR. Sans perte de généralité, si l'on appelle $n_0$, $n_1$, ..., $n_{99}$, les 100 nœuds du graphe, il suffit alors de considérer tout nœud $n_k$ comme nœud émetteur de trafic CBR vers le nœud $n_{n\text{-}k\text{-}1}$ où $k \in \{0, ..., 49\}$ au débit $r$. Ainsi 50 générateurs de trafic CBR ont été créés et 50 connexions ont été établies.

Afin d'apprécier ce que peut représenter un *réseau modérément chargé*, nous déterminons d'abord ce que représente un réseau saturé ensuite nous le réduisons arbitrairement à $\frac{2}{3}$. Nous avons déterminé cela dans le réseau défini précédem-

14. CBR est moins réaliste qu'un trafic suivant un processus de Poisson, mais permet, parce qu'il n'introduit pas d'aléa supplémentaire, de diminuer considérablement le nombre d'événements durant chaque simulation pour des petits écarts type. En effet, une exécution d'une simulation peut durer un temps entre 30 et 45 mn avec CBR, elle peut durer dix fois plus de temps avec d'autre modèles de trafic.

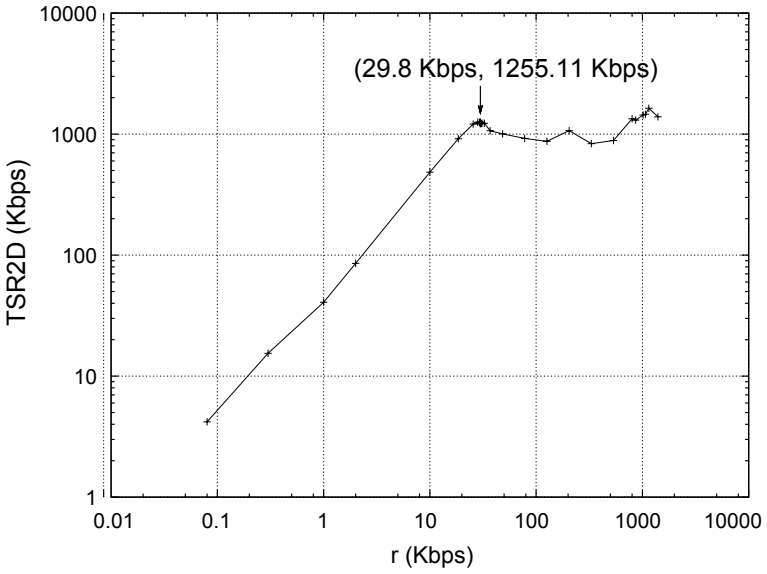


FIGURE 7.6 – Détermination de la saturation du réseau : TSR2D vs. $r$

ment mais ne contenant pas de trafic d'authentification.

Pour $r = 0.08$ Kbps, nous avons obtenu un volume total de données reçues par seconde et par l'ensemble des destinations (en anglais *Total per Second Received Data at Destinations* ou TSR2D) de 4.19 Kbps. Pour $r = 1400$ Kbps, TSR2D était aux alentours de 1390 Kbps. Pour $r = 700.04$ Kbps, TSR2D était d'environ 900 Kbps. Pour $r = 350.02$ Kbps, la valeur du TSR2D était proche de la précédente valeur. Les résultats sont illustrés par la figure FIGURE 7.6. Une poursuite fastidieuse des simulations et un report des résultats en faisant appel à la méthode de dichotomie permet de déterminer directement la valeur de $r$ pour laquelle il ya saturation du réseau. Nous avons obtenu $r = 29.88$ Kbps. TSR2D n'augmente plus à partir de cette valeur et les performances du réseau se dégradent. Nous prenons donc comme valeur de $r$ pour un réseau modérément chargé $\frac{2}{3}$ de cette valeur soit $\boldsymbol{r}$=**19.7 Kbps** [15].

## 7.5.2 Paramétrage du trafic AAA d'authentification

### 7.5.2.1 Paramétrage du protocole

Nous supposons que les phases d'inscription et d'initialisation ont eu déjà lieu. Les nœuds connaissent donc déjà les adresses IP des pairs AAA, les paramètres $n$ et $th$ ainsi que $th_M$, $th_i$, T, K et $t_{max}$ (cf. tableau 7.5). Nous supposons aussi que la phase de construction du réseau a eu déjà lieu et que l'on est donc en régime stationnaire.

Concernant les valeurs des seuils $th$, $th_M$ et $th_i$, nous utilisons les valeurs choisis à la section 7.2.3 (cf. tableau 7.2). Lorsqu'un nœud arrive dans le réseau,

15. On pourra trouver modeste ce débit pour un réseau dont les nombreux liens supportent un débit de 11Mbps. En effet, le réseau transmet au total à peu près $30 \cdot 50 = 1500$Kbps. Nous en recherchons la raison dans les conflits engendrés par la quantité de nœuds à portée les uns des autres. Une étude approfondie basée sur la notion nouvelle de mode et de mode maximum dans un graphe est en cours.

il choisit $th_M$ pairs parmi les 20 disponibles et les sollicite pour être authentifié et autorisé (cf. section 4.4.3 du chapitre 4). En cas d'issue favorable, il se voit attribuer un jeton d'accès de durée limitée que nous fixons arbitrairement à une heure. Cela correspond à une utilisation du réseau pendant un temps court, ainsi qu'il a été prévu dans les applications futures des MANETs (cf. section 1.3.2 du chapitre 1). Le client doit renouveler sa demande d'authentification au bout de cet intervalle. Cela amène à fixer comme durée de simulation une valeur légèrement supérieure à 1h (3600 s), par exemple 4000 s, soit 01:06:40 (cf. tableau 7.6). La longueur de cette durée et le caractère stationnaire des distributions sont suffisants pour garantir l'ergodicité du processus [139] [146] .

Les valeurs de K et de $t_{max}$ seront déterminées grâce aux simulations. D'abord, on fixe la valeur de K à 1 c.-à-d. qu'un client ne lancera qu'une seule tentative par demande d'authentification. Cela permettra de mesurer, dans un premier temps, le taux de réussite et de confronter les résultats trouvés à ceux prévus par la section 7.2.3. Le taux de réussite expérimental, $\tau$, se calcule à l'aide de la formule définie à la section 6.5.3 du chapitre 6, soit :

$$\tau = \frac{nombre\ d'authentifications\ abouties}{nombre\ d'authentifications\ lancées}$$

De plus, il sera possible de mesurer le temps nécessaire à la terminaison d'une authentification en examinant celles qui se sont conclues favorablement. A cette fin, nous tracerons la courbe des authentifications cumulées ce qui aura comme avantages supplémentaires d'illustrer le taux de réussite et de détecter, s'il y en a des phénomènes particuliers dans le réseau. Elle donnera, par ailleurs, pour ceux qui le souhaiteraient, le moyen d'appliquer le test de Kolmogorov-Smirnoff[16] en cas de besoin.

#### 7.5.2.2 Paramétrage du trafic

Finalement, tout a été fait pour que notre modèle décrive un univers stationnaire de nœuds se déplaçant régulièrement et relayant un trafic constant. C'est dans cet univers qu'intervient l'authentification des nœuds.

Dans le cas d'authentifications[17] régulières où les nœuds sont déjà établis dans le réseau, le trafic suit un processus poissonien. C'est le premier cas que nous examinerons. Pour chaque nœud client $\mathrm{JN_i}$, un intervalle de temps $d_i$ est tiré selon une loi exponentielle. Ainsi, le nœud $\mathrm{JN_1}$ lance sa demande à $t = d_1$, $\mathrm{JN_2}$ à $t = d_1 + d_2$, $\mathrm{JN_3}$ à $t = d_1 + d_2 + d_3, \cdots$, $\mathrm{JN_{80}}$ à $t = d_1 + d_2 + \cdots + d_{80}$. Afin qu'avec une probabilité suffisante, le nœud $\mathrm{JN_{80}}$ lance sa demande d'authentification à une date dont l'espérance soit bien inférieure à 4000, on fixe le paramètre de la loi exponentielle à 45s.

Imaginons, d'autre part, l'heure d'ouverture d'une grande société, celle de l'ouverture des magasins d'un quartier commerçant. En quelques dizaines de minutes, ce sont plusieurs centaines de nœuds qui vont s'authentifier. Cette situation peu compatible avec le modèle stationnaire, que permet de rendre un

16. Le test de Kolmogorov-Smirnoff permet de vérifier que la courbe des fréquences cumulées n'est pas en contradiction avec une fonction de répartition connue et, par conséquent, de voir si les temps obtenus peuvent être utilement modélisés par une variable aléatoire, par exemple gaussienne.

17. On parle en général d'authentifications mais cela inclut aussi, s'il y a lieu, les réauthentifications.

processus poissonien d'intensité constante, est celle que nous allons modéliser dans un deuxième temps. Pour chaque nœud client $\mathrm{JN_i}$, une date $d'_i$ est tiré selon une loi exponentielle à laquelle il lance sa demande d'authentification. Pour une commodité de lecture, d'analyse et de compréhension du fichier *auth-Traces.tr* (cf. section 7.4.2.7) et du fichier de traces classique de NS-2 (cf. section 7.4.1), nous avons trié ces dates par ordre croissant. Sans perte de généralité, on les note $d'_1 \leqslant d'_2 \leqslant \cdots \leqslant d'_80$. Ainsi, $\mathrm{JN}_1$ lance sa demande à l'instant $d'_1$, $\mathrm{JN}_2$ à $d'_2$,$\cdots$, $\mathrm{JN}_{80}$ à $d'_{80}$. Afin que les 80 JNs lancent chacun au moins une demande d'authentification au cours de la simulation, on fixe le paramètre de la loi exponentielle à 600s, soit 10mn.

### 7.5.3 Scénarios

Finalement, les différentes catégories de choix dont nous allons étudier l'impact sont résumées au tableau 7.7. Il y a donc huit scénarios à étudier :

| catégorie 1 : choix des pairs | catégorie 2 : modèle de trafic | catégorie 3 : triplets de seuils |
|---|---|---|
| pairs les plus proches (p.p.p.) | distribution exponentielle (exp.) | ($th = 5$, $th_i = 8$, $th_M = 11$) |
| pairs aléatoires (p.a.) | processus de Poisson (poisson) | ($th = 10$, $th_i = 14$, $th_M = 19$) |

Tableau 7.7 – Choix pour les scénarios de simulation

1. scénario 1 (p.p.p./exp./th=5) :
   choix de pairs les plus proches, distribution exponentielle et ($th = 5$, $th_i = 8$, $th_M = 11$),
2. scénario 2 (p.p.p./poisson/th=5) :
   choix de pairs les plus proches, processus de Poisson et ($th = 5$, $th_i = 8$, $th_M = 11$),
3. scénario 3 (p.a./exp./th=5) :
   choix aléatoire de pairs, distribution exponentielle et ($th = 5$, $th_i = 8$, $th_M = 11$),
4. scénario 4 (p.a./poisson/th=5) :
   choix aléatoire de pairs, processus de Poisson et ($th = 5$, $th_i = 8$, $th_M = 11$),
5. scénario 5 (p.p.p./exp./th=10) :
   choix de pairs les plus proches, distribution exponentielle et ($th = 10$, $th_i = 14$, $th_M = 19$),
6. scénario 6 (p.p.p./poisson/th=10) :
   choix de pairs les plus proches, processus de Poisson et ($th = 10$, $th_i = 14$, $th_M = 19$),
7. scénario 7 (p.a./exp./th=10) :
   choix aléatoire de pairs, distribution exponentielle et ($th = 10$, $th_i = 14$, $th_M = 19$),

8. scénario 8 (p.a./poisson/th=10) :
   choix aléatoire de pairs, processus de Poisson et ($th = 10$, $th_i = 14$, $th_M = 19$),

### 7.5.4 Résultats

Chaque scénario a fait l'objet d'une simulation. Nous avons ensuite tracé, pour chacun d'entre eux, le nombre de messages d'authentification cumulés qui ont été envoyés dans le réseau par l'ensemble des 80 clients et durant la durée totale de la simulation. Sur la figure 7.7, on distingue clairement les trafic suivants un processus de Poisson de ceux de distribution exponentielle. Le premier

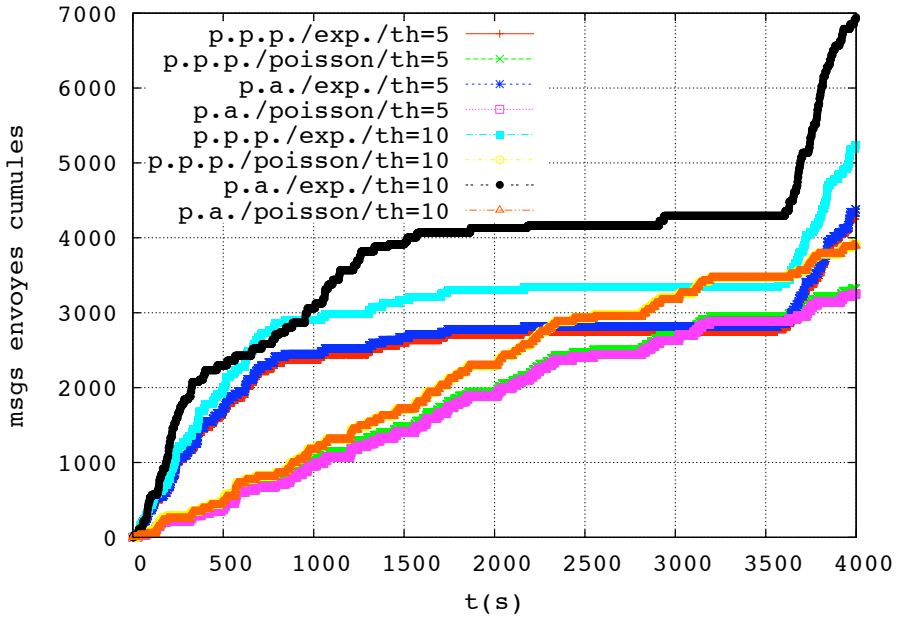


FIGURE 7.7 – Nombre de messages d'authentification envoyés cumulés

type étant régulier, les courbes sont presque linéaires (courbes verte, rose, jaune et orange). Dans le cas exponentiel, la majeure partie des messages d'authentification est envoyées pendant les 10 premières minutes (courbes rouge, bleue, bleu-verte et noire). Le trafic baisse ensuite de façon importante. Cela ressemble à un phénomène de *burst*. On voit, par ailleurs, que, vers la date $t = 3600$s, tous les nœuds qui se sont authentifiés au-début de la simulation, demandent à être ré-authentifiés, ce qui crée à nouveau un *burst*. Les authentifications se trouvent donc, à chaque *burst*, en compétition et la charge du réseau augmente considérablement en peu de temps. Cela est d'autant plus important que le seuil cryptographique est grand (courbes bleu-verte et noire où $th = 10$) et que le choix des pairs se fait de façon aléatoire (courbe noire).

La figure 7.8 montre les courbes du ratio d'authentifications réussies cumulées (RARC) (cf. section 7.5.2.1) pour chacun des scénarios ci-dessus. Le RARC étant le rapport entre le nombre d'authentifications réussies cumulées et le nombre total d'authentifications cumulées. En abscisse, on lit le temps

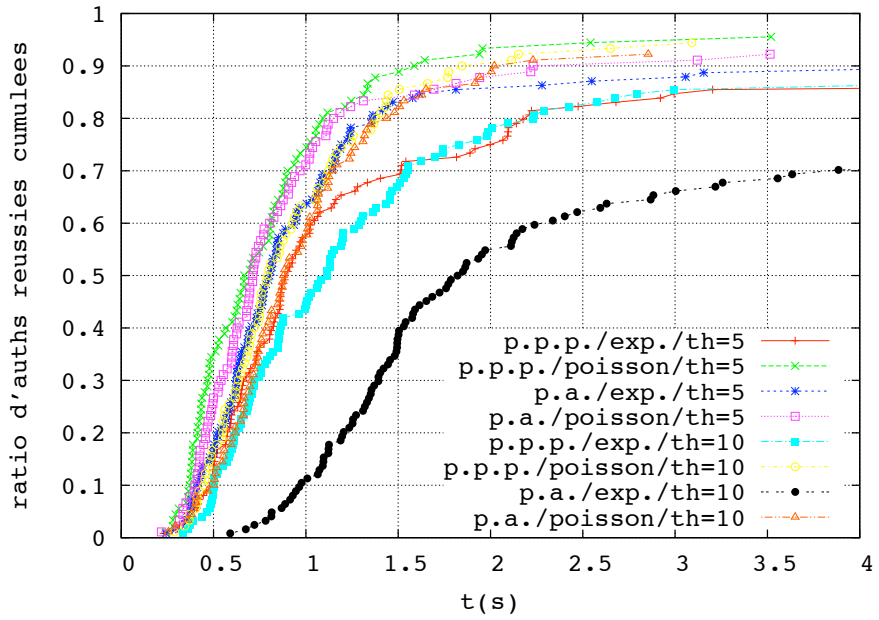


FIGURE 7.8 – Authentifications réussies cumulées

d'authentification (pour les authentifications réussies) et en ordonnée le taux de réussite.

On constate que le taux de réussite et le temps d'authentification sont meilleurs lorsque le processus des authentifications est stationnaire (processus de Poisson) que dans le cas exponentiel : courbe verte au dessus de la rouge, rose au-dessus de la bleue, jaune au-dessus de la bleu-verte et orange au-dessus de la noire. Pour les quatre cas où le processus est poissonien, 90% ou plus des nœuds ont réussi à s'authentifier en moins de 2s (dans le scénario p.a./poisson/$th = 5$ : 88%). D'autre part, ils sont entre 85% et 88% à s'authentifier en moins de 3s sauf dans le scénario p.a/exp./$th = 10$ qui donne manifestement les plus mauvais résultats : 70% des nœuds réussissent à s'authentifier en moins de 4s.

On remarque aussi qu'ils sont meilleurs, en général mais pas tout le temps, lorsque les pairs sont choisis parmi les plus proches que lorsqu'ils sont choisis aléatoirement : courbe verte au-dessus de la rose, bleu-verte au-dessus de la noire, jaune au-dessus de l'orange mais rouge au-dessous de la bleue.

En ce qui concerne les deux triplets de seuils étudiés, nous remarquons que, globalement, les performances sont meilleures pour le premier triplets c.-à-d. ($th = 5$, $th_i = 8$, $th_M = 11$) que pour le deuxième c.-à-d. ($th = 10$, $th_i = 14$, $th_M = 19$) : les courbes rouge et bleu-verte sont entrelacées, la verte est au-dessus de la jaune, la bleue au-dessus de la noire et la rose pour plus de 80% au-dessus de l'orange.

La figure 7.9 donne une première explication aux résultats donnés par la figure 7.8. On remarque que l'ordre des couleurs des courbes a été globalement inversé : moins les performances sont bonnes plus le nombre total de messages perdus est grand.

La figure 7.10 illustre, en bleu, le ratio d'authentifications échouées dès la

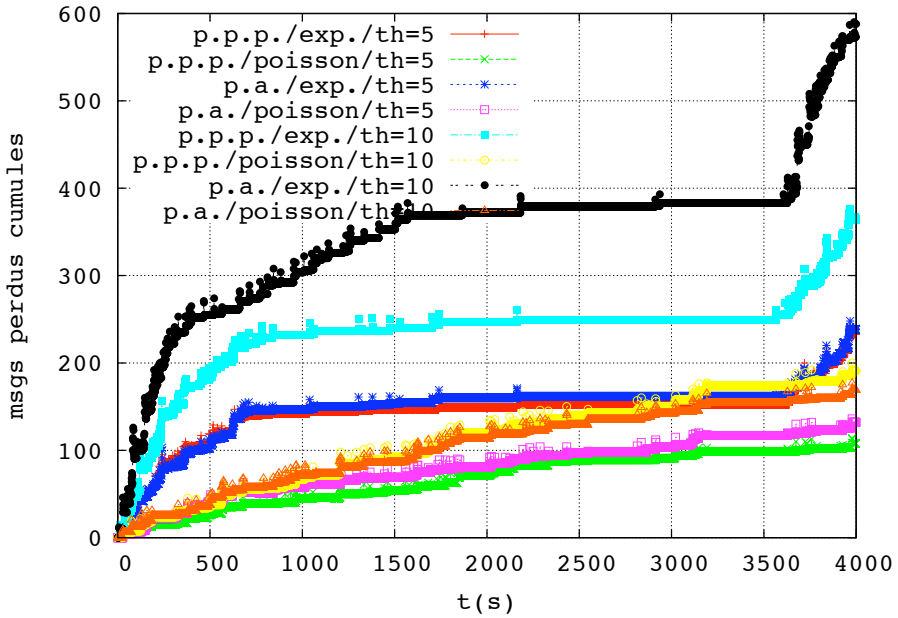


FIGURE 7.9 – Nombre de messages d'authentification perdus cumulés

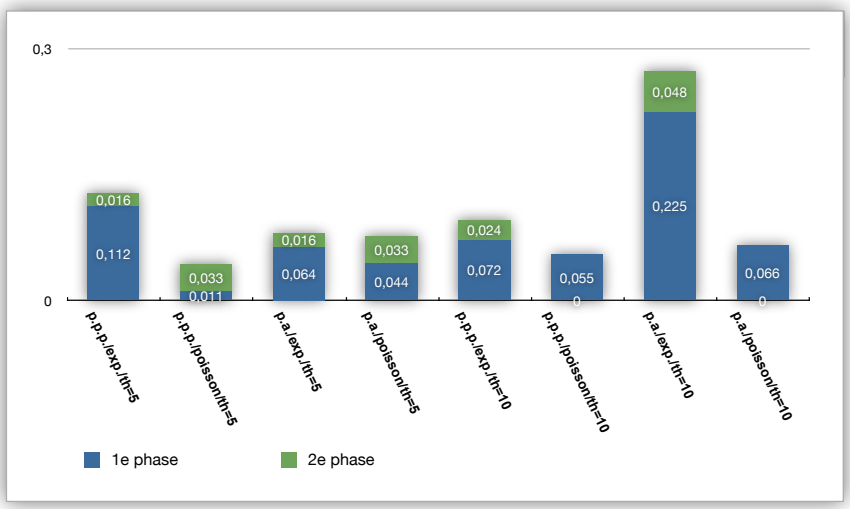


FIGURE 7.10 – Ratios d'authentifications échouées à la première phase et d'authentifications échouées à la deuxième phase

première phase défini comme le nombre d'authentifications échouées dès la première phase divisé par le nombre total d'authentifications qui ont été lancées.

En vert, elle montre le ratio d'authentifications échouées après réussite de la première phase et échec de la deuxième. Il est égal au rapport entre le nombre d'authentifications échouées à la deuxième phase divisé par le nombre totale d'authentifications qui ont été lancées.

On constate que la somme des deux, c.-à.-d le taux d'authentifications échouées, est le plus bas pour le scénario p.p.p./poisson/$th = 5$ (courbe verte sur la figure 7.8) et le plus haut pour le scénario p.a./exp./$th = 10$. Les taux d'authentifications échouées corroborent, bien entendu, les résultats illustrés par la figure 7.9, à savoir les nombres de messages perdus.

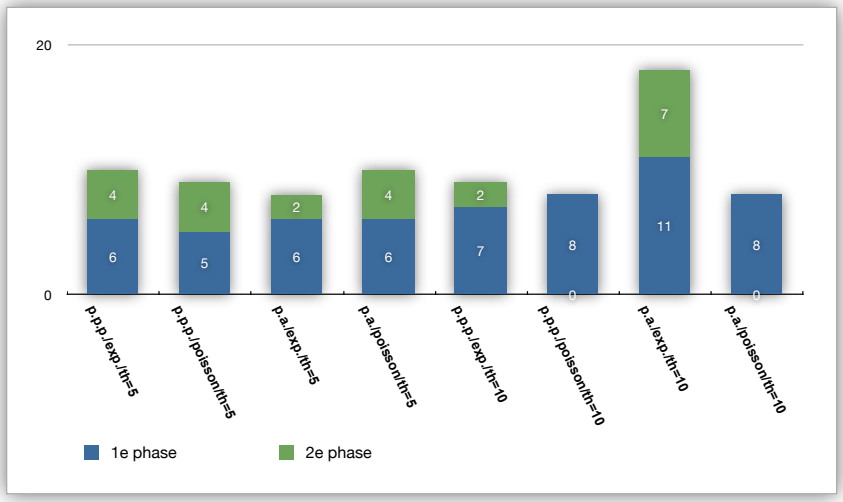


FIGURE 7.11 – Nombres moyens de pairs ayant répondu pendant une authentification échouée dès la première phase et pendant une authentification échouée à la deuxième phase

La figure 7.11 donne, pour chaque ratios de la figure 7.10, le nombre moyen de pairs qui ont répondus. En examinant à la fois le graphe de la figure 7.10 et celui de la figure 7.11, on note que, mis à part le scénario p.p.p./poisson/$th = 5$, une authentification échoue dès la première phase. De plus, cela arrive souvent alors que le nombre moyen de pairs qui ont répondu est supérieur ou égal au seuil cryptographique (voir les quatre premières et l'avant dernière barres de la figure 7.11). Ce constat nous a amené à proposer une approche qui touche à la conception même du protocole et grâce à laquelle nous espérons améliorer le taux de réussite. C'est l'approche d'*entrelacement* des messages de la première et de la deuxième phase (cf. section 7.7).

Par ailleurs, nous avons vu à la figure 7.8 que le taux de réussite était autour de 90%, sauf pour le cas exceptionnel p.a./exp./$th = 10$, ainsi que l'étude théorique l'avait prévu à la section 7.2. D'ailleurs, afin de valider cette étude lorsque $th = 5$, nous avons lancé une simulation par valeur de $th_M \in \{7, \cdots 17\}$ et de $th_i \in \{5, \cdots, th_M\}$ dans le cas où les pairs sont choisis aléatoirement et

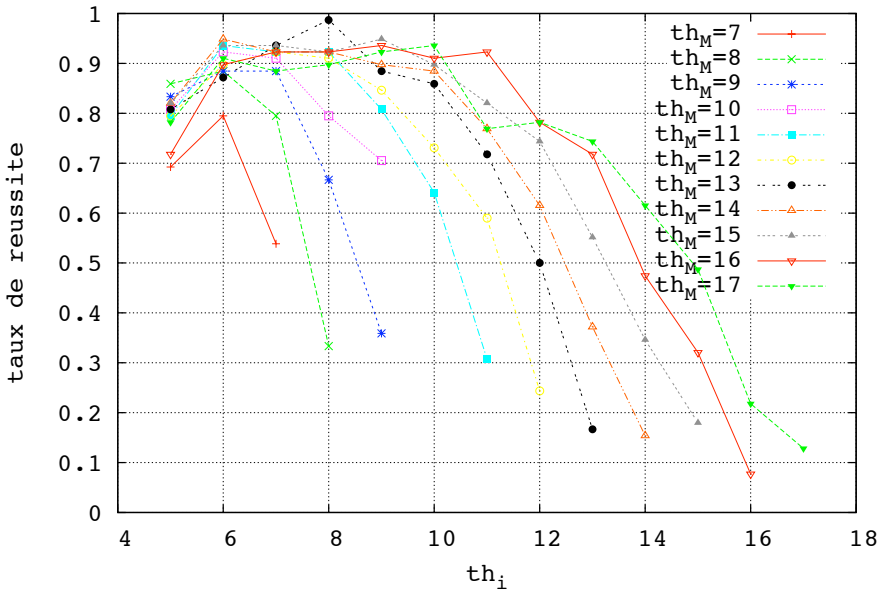


FIGURE 7.12 – Taux de réussite

le trafic d'authentification est distribué selon une loi exponentielle. Grâce à ces simulations, nous avons calculé le taux de réponse d'un pair AAA comme le rapport entre le nombre total de réponses reçues de tous les pairs et le nombre total de requêtes envoyées par tous les clients sur l'ensemble des valeurs de $th_i$ et de $th_M$. C'est la valeur de l'estimation de la probabilité de réponse d'un pair AAA, $\hat{p}$, que nous avons employée à la section 7.2.3 (cf. tableau 7.1).

Les résultats de ces simulations sont présentés sur la figure 7.12. On remarque que, globalement, pour les faibles valeurs de $th_i$ et de $th_M$, le taux de réussite est meilleur dans les simulations que dans l'étude théorique. En revanche, il est moins bon pour les fortes valeurs de $th_i$ et de $th_M$. Cela est dû à la valeur de $\hat{p}$ qui n'est pas identique pour tous les couples $(th_i, th_M)$. Nous avons vu, en effet, que le nombre de messages perdus augmente quand le nombre de pairs sollicités croît (cf. figure 7.9 lorsque on passe du triplet ($th = 5$, $th_i = 8$, $th_M = 11$) au triplet ($th = 10$, $th_i = 14$, $th_M = 19$)). Il serait donc utile de prendre en compte ce facteur pour affiner l'étude théorique.

Nous sommes, d'autre part, conscient qu'il faudrait augmenté le nombre d'échantillons, donc le nombre de simulations, pour avoir des résultats plus probants. Néanmoins, la longueur du temps d'exécution de ces simulations ne nous a pas permis de le faire.

## 7.6 Réglages de l'algorithme à multiples tentatives d'authentification

### 7.6.1 Estimation du temps d'attente maximum $t_{max}$

#### 7.6.1.1 Calcul de $t_{max}$ et validation de la probabilité de réussite

Considérons d'abord que les processus d'authentification sont exécutés sans l'application de l'algorithme à multiples tentatives d'authentification donc en un seul échange. Les simulations, lorsque les pairs sont choisis aléatoirement et le trafic est exponentiel, montrent que l'estimation expérimentale de la probabilité de réussite c.-à-d. le taux de réussite est aux alentours de 91% pour $th = 5$, ce qui est très proche de l'estimation analytique c.-à-d. 92.8% (cf. section 7.2.3). En revanche, il est d'environ 72% pour $th = 10$, ce qui n'est pas très proche de l'estimation expérimentale à savoir 92%. La raison est liée à la rafale de réponses arrivant au nœud sollicitant et dont l'impact est d'autant plus grand que $th$ augmente. Cela se traduit par un taux de collision plus important par paquet AAA envoyé (cf. la dernière colonne du tableau 7.9) engendrant une décroissance de la probabilité $p$ de réponse des pairs. De plus, le taux de réussite pour $th = 10$ est algébriquement plus sensible à l'estimation de $p$ que le taux de réussite pour $th = 5$. En effet si $p = 0.8$ au lieu de 0.86 comme cela était le cas à la section 7.2.3, il serait égal à 72% !

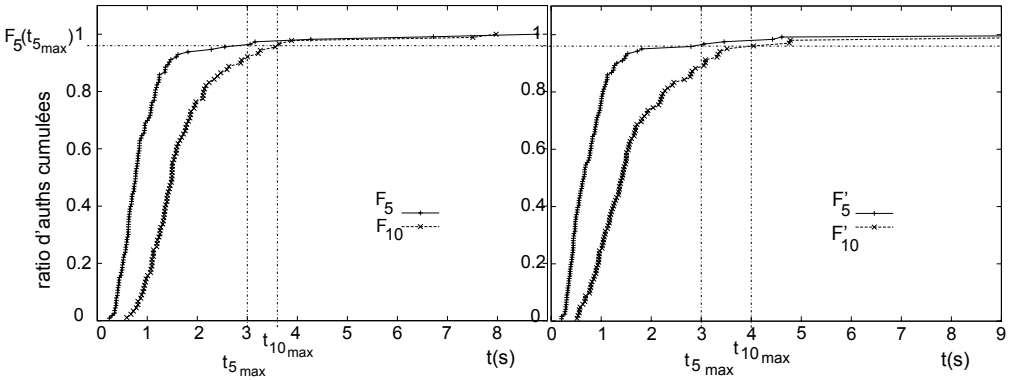


FIGURE 7.13 – Cumulated AAs for $th = 5$ and $th = 10$. On the left-hand side : *separation case*. On the right-hand side : *interleaving case*

D'autre part, on note $F_5$ et $F_{10}$ les fonctions de distribution cumulatives [18] du temps d'authentification pour respectivement $th = 5$ et $th = 10$. Leurs formes sont illustrées par la parties gauche de la figure 7.13. On peut raisonnablement estimer $t_{max}$ comme le temps maximum où 96% des authentifications réussies ont été abouties, ensuite $t_{max} = t_{5_{max}} = 3$ s pour $th = 5$ et $t_{max} = t_{10_{max}} = 3.8$ s pour $th = 10$ (c.-à-d. $F_5(t_{5_{max}}) = F_{10}(t_{10_{max}}) = 0.96$). Ces valeurs, nous les avons retenues et utilisées dans les simulations présentées dans les sous sections suivantes et où l'algorithme à multiples tentatives d'authentification a été

18. Là encore, on pourrait reprendre ces courbes et utiliser un test de Kolmogoroff-Smirnoff pour vérifier que les distributions obtenues correspondent à des lois connues en vue de conduire par la suite une utilisation analytique de ces lois. Nous ne nous sommes pas engagés dans cette voie.

exploité.

### 7.6.2 Réglage du nombre maximum de tentatives $K$

| seuils \ nbr de tentatives | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| $th$ =5, $th_M$=11, $th_i$=8 | 0.847 | 0.105 | 0.016 | 0.016 | 0.016 |
| $th$ =10, $th_M$=19, $th_i$=14 | 0.613 | 0.137 | 0.145 | 0.089 | 0 |
| seuils \ nbr de tentatives | 6 | 7 | 8 | 9 | 10 |
| $th$ =5, $th_M$=11, $th_i$=8 | 0 | 0 | 0 | 0 | 0 |
| $th$ =10, $th_M$=19, $th_i$=14 | 0 | 0 | 0.008 | .008 | 0 |

TABLEAU 7.8 – Fraction d'authentifications supplémentaires abouties en fonction du nombre de tentatives supplémentaires

Étant donnée $P_2$, la probabilité d'aboutissement d'une authentification comme elle a été définie à la section 7.2.3, et $F$, la fonction de distribution cumulative du temps d'authentification pour un triplet ($th$, $th_i$, $th_M$), la probabilité qu'une tentative d'authentification échoue après un temps d'attente de $t_{max}$ secondes est :

$$Q = 1 - P_2\, F(t_{max}) \tag{7.6}$$

et la probabilité d'échec après $K$ tentatives est : $Q^K$.

$lim\, Q^K = 0$, cela signifie que si un JN lance un nombre suffisant de tentatives, il réussira inéluctablement à s'authentifier. D'après les simulations, les valeurs $K = 5$ pour $th = 5$ et $K = 10$ pour $th = 10$ sont largement suffisantes pour atteindre 100% de réussite. Le tableau 7.8 montre la fraction d'authentifications supplémentaires qui aboutissent à chaque nouvelle incrémentation de $K$.

## 7.7 L'entrelacement des phases est-il une amélioration ?

L'approche initiale sépare la première et la deuxième phases (cf. section 4.4.3 du chapitre 4). Avec l'introduction des seuils $th_i$ et $th_M$, on a vu qu'un JN doit attendre au moins $th_i$ réponses des pairs AAA avant de pouvoir initier la deuxième phase. Une autre approche consiste à entremêler les deux phases en faisant en sorte que le JN envoie un message à chaque pair qui vient de répondre sans attendre l'arrivée de tous les $th_i$ réponses de $th_i$ pairs. Cela est possible à faire car les deux phases du protocole sont réalisées entre le JN et $th_M$ pair et non entre le JN et un seul pair. De plus il est très improbable que les réponses de deux pairs différents arrivent en même temps au JN.

On appelle *séparation* le cas où JN doit attendre toutes les réponses des pairs AAA durant la première phase avant d'initier la deuxième phase et *entrelacement* la situation où il envoie les messages de sa deuxième phase au fur et à mesure des réponses des pairs.

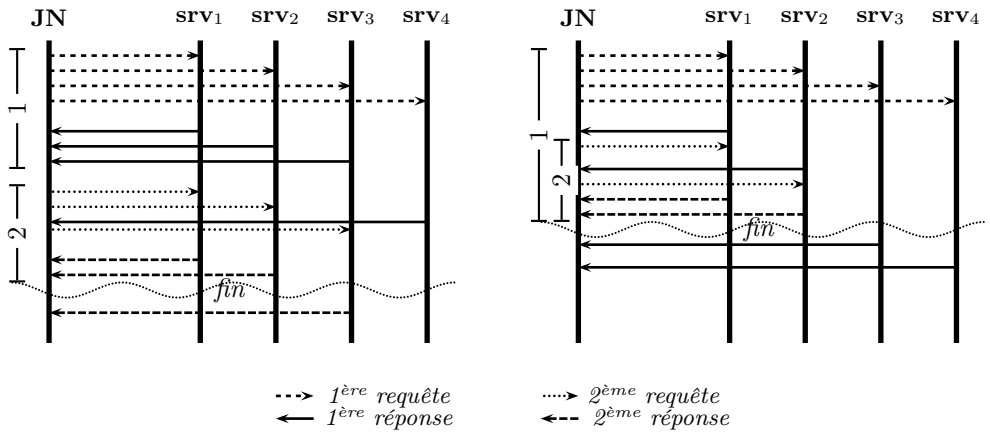


FIGURE 7.14 – $th_M = 4$, $th_i = 3$ and $th = 2$. A gauche : cas de *séparation*. A droite : cas *entrelacement*

La figure 7.14 montre que, dans le cas d'entrelacement, un processus d'authentification peut aboutir avec un nombre de réponses durant la première phase inférieur à $th_i$. Cela pourrait améliorer d'une part le temps d'authentification, donc le paramètre $t_{max}$, d'autre part le taux de réussite parce qu'un nombre total de messages moins important est suffisant pour accomplir une authentification.

### 7.7.1 Contribution de l'approche d'*entrelacement* par rapport à l'approche de *séparation*

| métrique / approche | $P_2$ | $t_{max}(s)$ | $K$ | msgs AAA envoyés par auth. | % de coll. par msg AAA |
|---|---|---|---|---|---|
| *séparation*, $th = 5$ | 0.91 | 3 | 5 | 39 | 63.84 |
| *entrelacement*, $th = 5$ | 0.98 | 3 | 4 | 35 | 69.25 |
| *séparation*, $th = 10$ | 0.72 | 3.8 | 10 | 78 | 75.18 |
| *entrelacement*, $th = 10$ | 0.82 | 4 | 8 | 71 | 83.40 |

TABLEAU 7.9 – Comparison of the *interleaving* vs *separation* approaches

Le taux de réussite et le nombre de tentatives ont été améliorés légèrement dans le cas d'*entrelacement* comparé au cas de *séparation* (cf. tableau 7.9). La figure 7.13 illustre par sa partie droite le profil de $F_5^{'}$ et de $F_{10}^{'}$, les fonction de distribution cumulatives du temps d'authentification pour respectivement $th = 5$ et $th = 10$ dans le cas d'*entrelacement*. On note que le temps d'authentification n'a pas été réellement amélioré. De plus, pour $th = 5$ (respectivement $th = 10$), le temps moyen d'authentification dans le cas de *séparation* est de 1.02 s (respectivement 2.28 sec) et il est de 0.93 s dans le cas d'*entrelacement*. Cette petite amélioration pour $th = 5$ et petite régression pour $th = 10$ peut

être expliquée par le fait que bien que le nombre de messages AAA par processus d'authentification a été généralement diminué dans le cas d'*entrelacement*, le nombre de collisions par messages AAA envoyé, donc par processus d'authentification a été en même temps augmenté (cf. tableau 7.9) : un message envoyé immédiatement par un JN après avoir reçu une réponse d'un pair a de forte chance de rentrer en collision avec les réponses arrivant des autres pairs ce qui augmente la contention.

## 7.8 Conclusion

Dans ce chapitre, nous avons proposé des lignes directrices pour trouver des réglages permettant d'optimiser les performances du protocole d'authentification. L'étude analytique a mis en évidence les choix possibles des valeurs des seuils pour un taux de réussite optimal. Cette étude pour un protocole à deux échange peut être appliquée à d'autres protocoles avec un nombre quelconque d'échanges. Il suffit pour cela de prendre autant de facteurs de type $P_1$ comme expliqué à la section 7.2.3. Les protocoles à échanges multiples peuvent être utilisés pour établir une relation de confiance en plusieurs étapes, selon le niveau de sécurité souhaité, entre un service distribué et un client. Lorsque les capacités des liens le permettent, de tels protocoles peuvent être utilisés pour échanger des données de volumes assez grand, comme des données multimédia, entre un service distribué et un client. Cela peut avoir lieu dans un MANET contenant plusieurs ordinateurs portables. Les résultats des simulations ont confirmé la probabilité de réussite prédite. Ces simulations ont aussi servi à fixer la valeur de l'ensemble des autres paramètres que nous avons définis dans le cadre de l'algorithme à multiples tentatives d'authentification.

Par ailleurs, l'entrelacement des deux échanges a amélioré légèrement le taux d'authentification et le nombre maximum de tentatives $K$ à fixer, mais il a amélioré le temps d'authentification de façon insignifiante.

Notre travail fournit une méthodologie pour définir les paramètres nécessaires à l'optimisation du taux d'authentification. Il donne des éléments pour améliorer le temps d'authentification à travers l'étude de l'entrelacement des échanges. Il permet aussi de définir le nombre de tentatives maximum d'authentification.

Cette approche est originale dans le sens où les performances et les valeurs des paramètres ont été étudiés dans un réseau où non seulement un trafic d'authentification a été injecté mais aussi où circulait un trafic ambiant simulant tout autres trafic de signalisation ou de données.

Nos résultats démontrent que les processus d'authentification lancés de façon régulière à travers le réseau ne chargent pas le réseau de plus de 1% du *throughput* total.

# Conclusion et perspectives

Au début de cette thèse, nous avons posé la question de l'antinomie de la mobilité et de la sécurité. Tout au long de ce travail nous avons montré qu'en ce qui concerne les réseaux ad hoc mobiles, appelés aussi MANETs, cette antinomie n'est qu'une antinomie de surface et qu'il est possible en y mettant beaucoup de soin de concilier ces deux optiques.

Dans un premier temps, nous avons étudié les réseaux ad hoc mobiles. Leur intérêt, initialement militaire, est en train de s'élargir au domaine civil et plus précisément au domaine commercial. Le but de cette étude était d'en comprendre les caractéristiques et le fonctionnement afin de proposer plus tard des solutions de sécurité adéquates. Nous avons vu que les défis qu'ils lancent sont de natures diverses. Les uns sont liés au caractère dynamique de la topologie de ces réseaux et à l'absence d'une infrastructure fixe et surtout administrée. Les autres proviennent du fait que l'autonomie du réseau pose la question de la participation effective des nœuds à ses opérations. Cela peut tenir à des raisons légitimes, au premier rang desquels se trouve la préservation au sein d'un nœud d'une autonomie énergétique suffisante ou celle du maintien d'une puissance de calcul ou d'une capacité de stockage locale convenable.

Nous avons ensuite recensé les méthodes propres à assurer la sécurité dans les réseaux les mieux connus, les réseaux filaires, et étendu nos investigations aux réseaux sans-fil de pareille structure. Traditionnellement, des techniques de contrôle d'accès conjuguées à l'usage d'outils cryptographiques sont employés pour sécuriser l'ensemble de ces réseaux, souvent rassemblés sous le nom de réseau à infrastructure. La sécurité repose alors sur des relations de confiance préalablement établies. Elle est grandement facilitée par la possibilité de mettre en place des points de contrôle à passage obligé. Quoiqu'il en soit, ces solutions ne se laissent pas aisément transposer au monde des réseaux mobiles sans infrastructure que sont les MANETs.

Avant de proposer une solution pour contrôler l'accès aux divers services potentiellement fournis dans un MANET à usage commercial, nous avons procédé à une taxonomie des techniques qui ont déjà été proposées dans ce cadre. Il en est apparu qu'une grande partie d'entre elles reposent sur la présence d'une administration permanente d'un ou d'une agglomération d'opérateurs. Nous avons appelé la catégorie des MANETs ainsi organisés, catégorie des MANETs tributaires. D'autre part, les solutions qui ont été imaginées dans le cadre d'un MANET autonome se sont souvent révélées encore immatures.

Nous avons donc décidé, dans un deuxième temps, de concevoir une solution pour mettre en place des services AAA dans un MANET partiellement tributaire. Ce choix nous a semblé logique puisque, dans un cadre commercial, l'intervention d'un tiers de confiance (l'opérateur) est nécessaire pour initialiser les clés

de sécurité et les politique de contrôle d'accès et préserver ainsi ses intérêts financiers, mais que cette intervention n'est plus nécessaire une fois les nœuds inscrits et les droits à utilisation distribués.

Plus particulièrement, en utilisant les adresses générées cryptographiquement (CGA) et la cryptographie à seuil, nous avons établi qu'un système d'autorité partagée formé d'un nombre restreint de pairs AAA pouvait, au cours d'un *processus d'authentification*, distribuer un jeton d'accès à un impétrant. La technique de la spécification formelle nous a permis de nous assurer que cette distribution se fait de façon sûre. A l'occasion de ces propositions, nous avons développé une technique originale de seuils intermédiaires qui peut s'utiliser lors de la mise en place de tout protocole distribué à échanges multiples en environnement instable.

L'extension du protocole SEND au-delà du voisinage, nous a ensuite permis, dans un *processus d'autorisation*, de garantir que toute utilisation ultérieure de ce jeton ne peut se faire que par le nœud qui vient d'être ainsi authentifié et par lui seul. Nous sommes ainsi certain de l'impossibilité du vol de droit d'accès.

Nous avons, par la suite, vérifié que le système de distribution proposé répondait avec une probabilité suffisamment élevée et un temps de réponse suffisamment bref pour que la méthode proposée puisse faire l'objet d'une implémentation pratique. Cela a nécessité d'une part le développement de calculs de probabilité suffisamment complexes pour que le recours à un système de calcul formel s'avère nécessaire. Pour aller plus loin dans notre étude, nous avons mis en place des simulations extrêmement soignées afin de parer aux critiques bien connues auxquelles les modèles de mouvement habituels ont donné naissance.

Comme le nouvel arrivant peut utiliser son jeton de façon sûre, tout membre du réseau peut de façon confiante dans l'identité d'un nœud, effectuer pour son compte les opérations demandées, qu'il s'agisse d'un simple routage ou de la fourniture de service ou de contenus, sous la seule condition d'un accord préalable. Ainsi il devient possible, sur la base de l'utilisation du jeton, de facturer l'ensemble des services accordés en étant certain de l'identité de celui qui les a requis.

## Perspectives de recherche

L'intérêt renouvelé porté par l'industrie à ce type de réseau étant en lien direct avec cette problématique de la fourniture de services via un Manet, nous pensons avoir fait un travail tant théorique que pratique d'intérêt.

Pour autant, diverses difficultés subsistent qui débouchent sur des domaines actuellement objets de recherche :

- comment s'assurer que les nœuds ainsi authentifiés coopèrent effectivement ? pour utiliser le langage des juristes qu'ils sont animés par l'*affectio societatis*.
- peut-on faire en sorte que des équipements disposant de faibles ressources puissent jouer le rôle de pair AAA malgré la consommation de ces ressources impliquée par l'utilisation de la cryptographie à seuil ?
- comment collecter les comptes rendus d'utilisation de jetons éparpillés au sein du réseau et redondants pour les synthétiser en une véritable comptabilité en vue de rendre un service AAA complet ?

Sur le premier point, on sait que des recherches sont en cours utilisant la théorie

économique et considérant chaque nœud comme un *agent autonome* en vue de mettre en place un système incitatif de pénalité et de récompense efficace.

Sur le deuxième point, la communauté scientifique travaille depuis déjà quelques années sur l'utilisation d'une cryptographie basée sur les courbes elliptiques. Celle-ci permet en effet, au moins en théorie, d'utiliser avec une égale force de résistance à la cryptanalyse des clés plus courtes demandant moins d'opérations cryptographiques pour un résultat équivalent. L'intégration d'un tel système ou même d'un système basé sur les courbes hyperelliptiques (SURF 1271 [147]) ou sur des corps autres que $\mathbb{Z}_p$ (XTR) peut ainsi permettre des économies d'énergie. Cela nécessite une étude aussi bien théorique, c'est-à-dire portant sur le source des algorithmes, que pratique, c'est-à-dire portant sur les instructions machine effectivement exécutées qui restent à notre connaissance à mener.

Sur le troisième point, des recherches sont actuellement en cours au sein de notre laboratoire. Il s'agit, dans un premier abord, de définir quels nœuds du réseau sont concernés par la collecte des informations sur l'utilisation des jetons lors des consommations des services souscrits. Le format de ces informations reste à préciser et le problème éventuel de leur perte en cas de départ de nœuds collecteurs doit être envisager avec sérieux. Comme, d'autre part, l'envoi de ces informations à une entité centrale sous contrôle de l'opérateur est indispensable, quel protocole devons nous utiliser et comment nous assurer de la cohérence des informations provenant de diverses sources collectrices ? Ce sont autant de questions qui ne cessent d'en amener d'autres.

Naturellement des recherches nécessitent des outils adaptés. C'est ainsi que nous envisageons de nous doter d'un modèle de déplacement plus réaliste que le traditionnel RWP. L'ensemble des déplacements doit pouvoir s'approcher au plus près de ce que l'on observe dans un environnement réel structuré. La maîtrise d'outils de simulation récents paraît désormais inéluctable. La détention d'un savoir faire approfondi sur les techniques de preuves formelles apparaît comme une nécessité prometteuse. Ces directions d'acquisition de compétence au sein de l'équipe apparaissent comme le pendant nécessaire de développement dans les directions de recherche que nous venons d'indiquer.

# Annexe A

# AODV

AODV comme OLSR sont les deux premiers protocoles de routage à avoir fait l'objet de RFC depuis 2003. OLSR est plus adapté aux environnements de mobilité réduite que AODV et *vice versa*. C'est donc pour le deuxième que nous avons attaché le plus d'attention pour l'instant car nous nous intéresserons dans la deuxième partie à un environnement d'assez forte mobilité.

D'après le RFC 3561 [12], AODV peut opérer dans un réseau ad hoc mobile comprenant de 10 à 1000 nœuds. Il suppose que, pour toutes les opérations, tous les nœuds sont bienveillants. Ses messages sont transportés dans des messages UDP destinés au port 654. Lorsqu'un nœud source $S$ veut joindre un nœud destinataire $D$, s'il n'a pas de route vers celui-ci, il déclenche un processus AODV de découverte de route. C'est ainsi que ce protocole est réactif i.e. qu'il n'établit pas de route tant qu'il n'y a pas de demande. $S$ diffuse alors un message RREQ (*Route Request*) dans le réseau demandant une route vers $D$. Le mécanisme se fait d'une manière distribuée i.e. que tout nœud $R$ du réseau ayant reçu un RREQ enregistre une route vers $S$, appelée *route inverse* (en anglais *reverse route*), en notant le nœud qui lui a relayé le RREQ comme le *prochain saut* (en anglais *next hop*) vers $S$. Le rôle d'une route inverse peut se révèler au moment où une réponse est envoyée à $S$ car celle-ci, contrairement au RREQ, doit être *unicast*. Pour cela, la durée de vie d'une route inverse est en général courte, elle est détruite si elle ne sert pas comme route à la réponse.

En plus d'enregistrer une route inverse, $R$ vérifie dans sa table de routage s'il n'a pas une route *assez récente* vers $D$. Si la recherche s'est révélée positive, il prend l'initiative de répondre au RREQ par un RREP (*Route Reply*) envoyé à $S$ sur la route inverse et contenant la route trouvée. Dans le cas contraire, $R$ diffuse à son tour le RREQ. S'il arrive à la destination $D$, celle-ci note une route inverse et envoie un RREP à $S$. Ainsi, le nœud $S$ peut recevoir plusieurs réponses, il en choisit alors celle qui décrit la route la plus récente. S'il en existe plusieurs, il choisit la plus courte d'entre elles i.e. celles qui a le minimum d'arêtes [1].

Par ailleurs, la construction de routes par un nœud vers ses voisins [2] peut

1. Par définition, la *longueur* d'une route est le nombre d'arêtes qui la compose i.e. le nombre de sauts nécessaires pour aller de la source à la destination. Ainsi, les adjectifs *longue* et *courte* exprime le fait qu'une route possède plus ou moins d'arêtes.

2. Dans le domaine de l'étude des graphes, la convention définit les nœuds se trouvant à un saut d'un nœud donné comme étant ses *voisins*. Parfois, on rencontre aussi dans la littérature l'expression *voisins immédiats* qui désigne la même chose.

avoir lieu à la réception de messages RREP particuliers, appelée messages Hello. Ces messages décrivent une route d'une seule arrête dont la destination est le nœud qui a envoyé le Hello lui même. Ce procédé sert à maintenir des informations récentes sur la connectivité dans le voisinage.

Comme tout protocole de routage, AODV maintient une table de routage au niveau de chaque nœud, mise à jour tout au long des échanges AODV à travers le réseau. Chaque table de routage comprend une entrée pour chaque route. L'aspect récent ou non d'une route, évoqué plus haut, est définit grâce à un numéro de séquence dont la gestion garantit, entre autre, l'absence de boucles dans cette route. C'est l'un des points forts de la conception de AODV. Les éléments enregistrés dans une entrée décrivant une route incluent l'adresse IP de la destination, son numéro de séquence, des drapeaux décrivant l'état de la route (valide, invalide, réparable, en train de réparation, etc), l'interface par laquelle la destination est joignable [3] , le nombre de sauts nécessaires pour l'atteindre, le prochain saut dans sa direction, la liste des précurseurs et le durée de vie de la route. Les précurseurs sont les nœuds voisins qui sont susceptible d'utiliser le nœud courant comme un prochain saut vers la destination considérée.

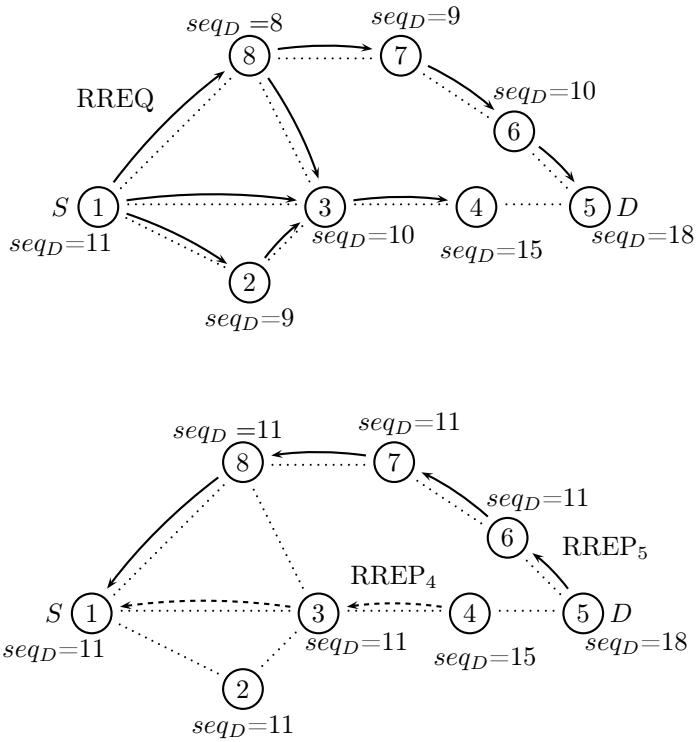


FIGURE A.1 – Envoi de RREQ par la source $S$ et réception de deux réponses RREP$_4$ du nœud 4 et RREP$_5$ du nœud 5

Comparer les numéros de séquence de la destination de plusieurs routes permet de les classer par ordre de récence : plus ce numéro est grand, plus

3. Parce qu'un nœud peut avoir plusieurs interfaces de réseau sans-fil voire même filaire, cette donnée permet de faire un choix judicieux de l'interface de réseau qui utilisera la route considérée.

la route est récente. La mise à jour du numéro de séquence d'une destination, figurant dans une entrée de la table de routage, a lieu chaque fois qu'un message AODV est reçu. Sa nouvelle valeur sera le maximum de sa valeur actuel et de celle figurant dans le champ *Destination Sequence Number* du message AODV. Par ailleurs, si le prochain saut vers cette destination n'est plus disponible, le numéro de séquence de celle-ci est augmenté de 1. La FIGURE 7.4.2.1 illustre ce fonctionnement : $S$ envoie une requête ayant comme numéro de séquence de la destination $D$ la valeur $seq_D = 11$, qui est supérieure à tous les numéros de séquence tenus par les autres nœuds sauf le nœud 4. Tous les nœuds mettent alors à jour la valeur de $seq_D$ dans leurs tables de routage sauf le nœud 4 qui entreprend d'envoyer une réponse $\text{RREP}_4$ à $S$. $D$ envoi aussi une réponse $\text{RREP}_5$ à $S$. Comme $D$ détient un plus haut numéro de séquence que le nœud 4, $S$ retient la route proposée dans $\text{RREP}_5$ bien qu'elle soit plus longue.

Toutefois, un nœud ne gère pas seulement les numéros de séquence des destinations, mais aussi son propre numéro de séquence. Ainsi, avant chaque nouvelle demande de route, il augmente celui-ci de 1 préalablement à la diffusion d'un RREQ. De plus, s'il se trouve être la destination d'une requête RREQ, avant de répondre par un RREP, il doit mettre à jour son numéro de séquence au maximum de sa valeur actuelle et de la valeur figurant dans le champ *Destination Sequence Number* du RREQ reçu.

L'emploi de numéros de séquences conjointement avec d'autres paramètres, comme le TTL (*Time To Live*), le nombre de sauts et les minuteurs (en anglais *timers*) apporte plusieurs avantages, tels que la limitation du nombre de messages diffusés à chaque nouveau RREQ généré, la connaissance du degré de récence d'une route, et aussi, comme nous l'avons mentionné, l'évitement de boucles dans une route [148]. AODV gère aussi les erreurs de transmission, les ruptures de liens, l'expiration de la durée de vie d'une route, la non réponse à une demande de route, etc. et définit pour cela un mécanisme utilisant des messages PERR (*Route Error*). Il déroule, par ailleurs, des étapes précises lors de l'initialisation des nœuds, tient compte des liens unidirectionnels, gère les interactions entre des sous-réseaux, dit réseaux *agrégées* (en anglais *Aggregated Networks*), opérant avec AODV et entre des réseaux utilisant AODV et d'autres, dit *extérieurs*, n'en utilisant pas.

# Annexe B

# Rappels de cryptographie

## B.1 Polynômes

Dans tout ce qui suit $\mathbb{K}$ désigne un corps quelconque, fini ou infini. Pour fixer les idées, ce peut être le corps $\mathbb{Q}$ des rationnels, le corps $\mathbb{R}$ des réels, celui $\mathbb{C}$ des complexes, ou un corps fini à $q$ éléments appelé *corps de Galois* (en anglais Galois field) et noté $\mathbb{GF}(q)$ où $q = p^k$, $p$ est un entier premier et $k$ un entier supérieur ou égal à 1. Rappelons que si $q$ est un entier premier $p$ (cas où $k = 1$), $\mathbb{GF}(p)$ n'est autre que l'anneau $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$ des entiers modulo $p$ qui est, dans ce cas, un corps.

On considère alors l'ensemble $\mathbb{K}[X]$ des polynômes en la variable $X$ et à coefficients dans $\mathbb{K}$, c'est à dire les quantités qui s'écrivent, pour un certain $n$ entier dépendant de $P$ : $P = \sum_{j=0}^{n} a_j X^j$ où $a_n \neq 0$. $n$ s'appelle alors le degré de $P$. On écrit $n = d^o(P)$. On écrit parfois $P(X)$ au lieu de $P$ pour rappeler que $X$ est la variable et $P(x)$ pour désigner l'expression obtenue en remplaçant $X$ par $x$ dans $P$. Si $P(a) = 0$ on dit que $a$ est une *racine* de $P$.

On sait que $\mathbb{K}[X]$ est un anneau euclidien, c'est-à-dire, pour faire court, que l'on sait ajouter soustraire et multiplier des polynômes et que cela donne des polynômes, que l'on sait simplifier des produits de polynômes, qu'il existe une division euclidienne pour deux polynômes $A$ et $B \neq 0$. En général un polynôme n'a pas d'inverse. Les règles de calcul sur les polynômes sont celles que nous connaissons depuis le lycée [149], étant entendu que lorsque $\mathbb{K}$ est un corps de Galois, un peu de soin est à prendre pour effectuer des calculs modulo $p$. On sait en particulier que seul le polynôme nul a un nombre de racine supérieur à son degré.

### B.1.1 Convention pour les produits de plusieurs facteurs

Rappelons d'abord qu'un produit de zéro facteur est égal à l'unité. Soit un produit de $n$ facteurs $\alpha_1\alpha_2\alpha_3\alpha_4...\alpha_n$. Pour écrire un produit en indiquant que l'on veut faire le produit de tous les facteurs sauf un, on écrit ce facteur surmonté d'un accent circonflexe, que l'on peut lire à voix haute *oublier* avant le facteur considéré. Ainsi on a :

$$\alpha_1\alpha_2\widehat{\alpha_3}\alpha_4...\alpha_n = \alpha_1\alpha_2\alpha_4...\alpha_n$$

On peut de même omettre plusieurs facteurs dans un produit en mettant un accent circonflexe sur chacun d'eux.

### B.1.2 Interpolation de Lagrange

Soient $(n+1)$ éléments donnés de $\mathbb{K}$, $x_0, x_1 \ldots, x_n$. On va leur associer $(n+1)$ polynômes $L_0(X), L_1(X), L_2(X), \ldots, L_n(X)$. Pour cela on procède en deux temps. Tout d'abord on définit

$$\ell_j(X) = (X - x_0)(X - x_1) \ldots (\widehat{X - x_j}) \ldots (X - x_n).$$

Par construction $\ell_j(X)$ est un polynôme de degré $n$ qui s'annule en chacun des $x_i$ autres que $x_j$ où sa valeur est non nulle. On définit alors $L_j(X) = \ell_j(X)/\ell_j(x_j)$. Il est alors clair que chacun des $L_j$ est un polynôme de degré $n$ qui s'annule en chacun des $x_i$ sauf en $x_j$ où il vaut 1.

Maintenant soient $(n+1)$ nombres $y_0, y_1, \ldots, y_n$ donnés, le polynôme

$$\mathcal{L}_n(X) = y_0 L_0 + y_1 L_1 + \cdots + y_n L_n$$

est de degré inférieur ou égal à $n$ et vaut $y_i$ en chaque $x_i$. Un tel polynôme est unique. En effet s'il existait un second polynôme ayant les mêmes propriétés, la différence entre ces deux polynômes serait un polynôme de degré inférieur ou égal à $n$ et aurait $(n+1)$ racines : $x_0, x_1, \ldots, x_n$. On sait alors que c'est le polynôme nul.

Le polynôme unique $\mathcal{L}_n$ dont nous venons de démontrer l'existence s'appelle le *polynôme d'interpolation de Lagrange* aux points $(x_i, y_i)_{0 \leqslant i \leqslant n}$ [149].

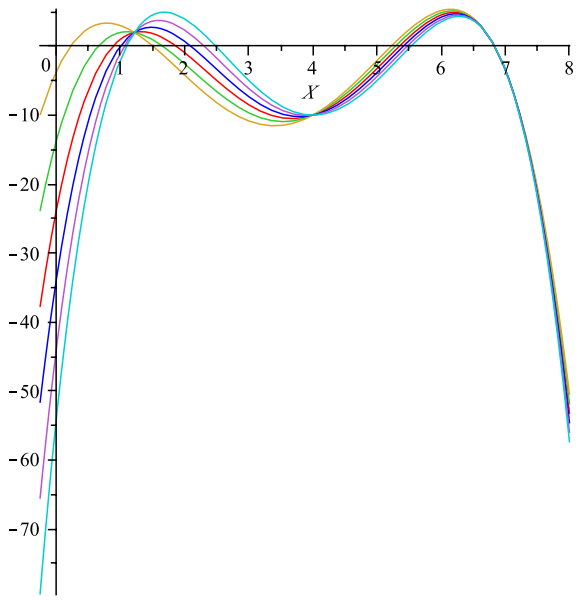


FIGURE B.1 – Tracés de $\mathcal{L}_5$ aux points $(1, y), (1.22, 2), (4, -10), (6.7, 2), (7.1, 6)$.

S'il est clair que la valeur en un point différent de tous les $x_i$ du polynôme d'interpolation de Lagrange dépend de chacune des valeurs, il est remarquable

que cette dépendance est sensible. La figure B.1 met cela en évidence : pour des valeurs de $y$ prises dans $(-2, -1, 0, 1, 2, 3)$ la valeur en 0 varie de quelques dizaines. Cette sensibilité de la valeur du polynôme de Lagrange en un point aux différentes autres valeurs est une propriété utile pour la suite.

## B.2 Groupes cycliques

Un groupe est un ensemble $G$ muni d'une loi de composition interne, notée par juxtaposition (le composé des éléments $a$ et $b$ se note $ab$), qui a les propriétés d'associativité, d'existence d'un élément neutre (noté parfois $e$), d'existence pour tout élément $a$ d'un symétrique $a^{-1}$. Une partie d'un groupe $G$ qui est elle même un groupe s'appelle un *sous-groupe*. Si $P$ est une partie d'un groupe $G$, le plus petit sous-groupe de $G$ qui contient $P$ est appelé le sous-groupe *engendré* par $P$ et noté $< P >$. Un théorème de Lagrange affirme que l'ordre, c'est à dire le nombre d'éléments d'un sous-groupe divise l'ordre du groupe dans lequel il est inclus.

Parmi les groupes, il en est de particulièrement utilisés en cryptographie, ce sont les *groupes cycliques*. Il s'agit des groupes ayant un nombre fini d'éléments et engendrés par une partie réduite à un seul élément $g$. On écrit un tel groupe $G =< g >$ à lire « $G$ est engendré par l'élément $g$ ». Alors les éléments du groupe sont l'élément neutre $e$, $g$, $g^2$, ..., $g^{N-1}$. Pour soutenir la réflexion, on peut imaginer les éléments du groupe rangés le long d'une courbe fermée dont un point particulier représente l'élément neutre, la multiplication par $g$ faisant passer d'un point au suivant le long de la courbe dont on dit qu'elle représente l'*orbite* de $g$ (cf. FIGURE B.2).

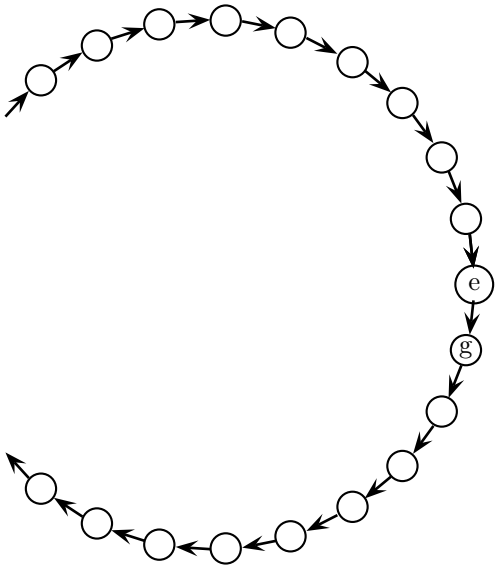


FIGURE B.2 – Un groupe cyclique s'identifie à l'orbite de $g$

On peut ainsi dire qu'un groupe cyclique représente conceptuellement une roue codeuse. Contrairement aux dispositifs matériels où la roue est de taille

limitée, ne serait-ce que pour des raisons d'encombrement, une telle « roue conceptuelle » peut-être immense.

### B.2.1 Exponentiation dichotomique

Dans un groupe $G =< g >$, on a souvent besoin de calculer $g^n$ comme fonction de $n$ pour des valeurs de $n$ qui peuvent aller jusqu'à l'ordre du groupe. Un calcul intuitif prendrait $n-1$ multiplications. En fait, il en faut en général beaucoup moins. En effet l'algorithme décrit en pseudo-code ci-dessous :

```
fonction puiss(g,n)
    if  n= 0  then return e
    elsif odd(n) then return g*puiss(g,n-1)
    else return puiss(g,n/2)^2
end
```

donne la valeur de la puissance en une multiplication par chiffre binaire de $n$ plus une multiplication par chiffre binaire égal à 1. Soit au plus $2 \times \log_2 n$ ce qui est beaucoup plus petit que $n$ lorsque $n$ est grand.

### B.2.2 Exemples utiles

Il est important de connaître un certain nombre de groupes usuels. Ces groupes sont utiles en ce sens que l'on détient une représentation particulière de leurs éléments, représentation adaptée à des calculs de produits efficients. En effet, pour les groupes d'une taille adaptée à la cryptographie moderne (de l'ordre de $10^{40}$ éléments) la représentation intuitive formée à partir de la notion d'orbite comme la liste (circulaire) des éléments du groupe est souvent compliquée à établir, matériellement impossible à stocker dans un ordinateur réel (ne pas oublier qu'un Tera est seulement $10^{12}$ et un Péta $10^{15}$) et même si cela était possible, peu propice aux calculs explicites.

Tout d'abord, si $\mathbb{K}$ est un corps fini (et l'on sait alors que la multiplication y est commutative d'après le théorème de Wedderburn [150]) l'ensemble des éléments non nul de $\mathbb{K}$, noté $\mathbb{K}^*$ est un groupe cyclique (pour la multiplication). Comme $\mathbb{K}$ a $p^k$ éléments, cela fournit un groupe à $p^k - 1$ éléments. Un exemple typique est $p = 2^{127} - 1 = 170141183460469231731687303715884105727$ dont on peut s'assurer , par exemple avec un logiciel adapté, qu'il est premier. Le groupe des inversibles $G = (\mathbb{Z}/p\mathbb{Z})^*$, qui a $(p-1)$ éléments contient la classe du nombre 5 qui est premier avec $2^{127} - 2$. Ainsi $G =< 5 >$ mais on a aussi $G =< 29 >$ et de même $G =< r >$ pour chaque $r$ pris parmi les 43877460003638844838728694809722093568 entiers premiers avec $2^{127} - 2$ qui ne représentent pourtant que 25% des éléments du groupe.

Les *courbes elliptiques* donnent aussi des groupes ayant des formes intéressantes. Disons tout de suite qu'une *courbe elliptique*, qui est de degré 3 n'est pas une conique, qui est de degré 2 et donc ce n'est pas une ellipse. Elle n'en a d'ailleurs même pas la forme. Son nom vient de l'utilisation qui est faite de ces courbes particulières pour calculer par intégration certaines longueurs liées aux ellipses mais ce point éclairci nous n'en parlerons plus.

Dans un repère convenable, une courbe elliptique $E$ est une courbe d'équation $(E) : y^2 = W(x) = x^3 + ax + b$, c'est à dire que c'est l'ensemble des points

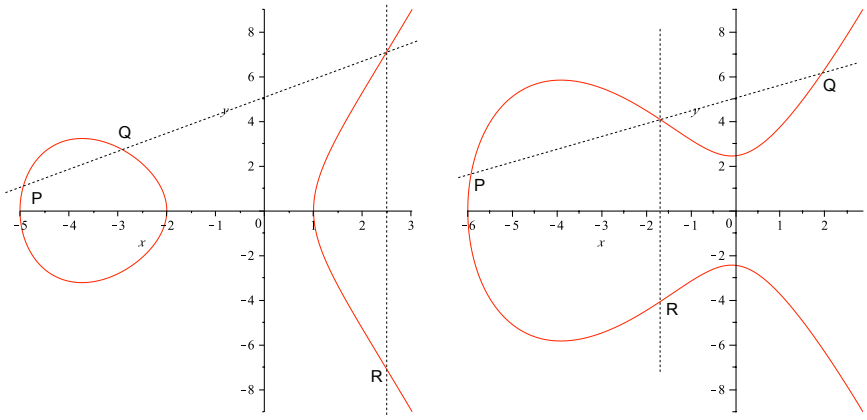


FIGURE B.3 – Loi de composition $R = P \cdot Q$ sur deux courbes elliptiques : A gauche $W$ a trois racines, à droite une seule

$(x, y) \in \mathbb{K}^2$ qui vérifient $(E)$. On y rajoute ordinairement un *point à l'infini* commun à toutes les droites parallèles à l'axe $Oy$ (cf. FIGURE B.3). En général, une droite qui coupe $E$ en deux points $P$ et $Q$ la coupe en un troisième point dont le symétrique par rapport à l'axe $Ox$ est un point $R$ noté $R = P \cdot Q$. Ceci donne une loi commutative qui munit les points de $E$ d'une loi de groupe (ce n'est pas immédiat à démontrer). On dispose ainsi d'un groupe qui a l'avantage d'une représentation commode, permettant des calculs efficaces, assez rapides. Dans les applications cryptographiques on prend les points de la courbe elliptique à coefficients entiers modulo un « grand » nombre premier. Un exemple « jouet » peut être trouvé à partir des valeurs modulo 29 des points de $y^2 = x^3 + x + 1$ qui donne un groupe à 40 éléments ((0, 26), (1, 0), (1, 12), (1, 17), (1, 29), (2, 14), (3, 11), (4, 24), (5, 10), (6, 13), (7, 6), (7, 25), (7, 27), (8, 19), (12, 8), (12, 22), (12, 28), (13, 16), (14, 18), (15, 18), (16, 16), (17, 8), (17, 22), (17, 28), (21, 19), (22, 6), (22, 25), (22, 27), (23, 13), (24, 10), (25, 24), (26, 11), (27, 14), (28, 0), (28, 12), (28, 17), (28, 29), (29, 26) , $\infty$).

## B.3 Tronçons et Codons

En général les systèmes de chiffrement représentent les données à chiffrer par des entiers inférieurs à un entier (très grand) fixé. Ils limitent ainsi la taille des messages susceptibles d'être chiffrés. Par exemple on peut imaginer envoyer des messages de 1024 bits (64 octets). Les messages usuels étant souvent plus longs la solution la plus courante consiste à découper le message original en tranches (ici de 64 octets) que l'on appellera *tronçons*. Le dernier tronçon est alors éventuellement complété à la même taille que les autres, par exemple en le concaténant avec un nombre tiré au hasard. Chaque tronçon est alors chiffré séparément pour donner un élément codé que nous appellerons *codon*. Le message transmis est alors la concaténation des codons. A la réception le message est décomposé en codons qui sont déchiffrés séparément restituant les tronçons que l'on n'a plus qu'à concaténer pour retrouver le message initial.

## B.4 Problèmes difficiles et fonctions à sens unique

Les bases de la cryptographie moderne ont été posées dans l'article de Diffie et Hellmann [85] qui mettent l'accent sur l'utilité des fonctions à sens unique. Il s'agit de fonctions, « faciles » à calculer, mais dont la réciproque est « très difficile »à calculer. Une première approche de ce type de fonctions peut se faire sur la base de la NP-complétude selon la thèse de Pierre-Alain FOUQUE [151]. Partant d'un ensemble fini, mais très grand $M$ de messages $m_i$, on construit pour chaque clé $k \in K$ une fonction $E_k$ bijective à valeurs dans un ensemble $C$ de codons $c_i$. On suppose que le calcul de $E_k(m)$ est pour tout $m$ un problème en temps polynomial. On dira que $E_k$ est à sens unique si le calcul de la réciproque $D_k$ de $E_k$ est non polynomial. Une telle approche suppose l'existence de telles fonctions. On ne dispose pas d'un résultat d'existence pour de telles fonctions et il reste à utiliser des fonctions dont *l'on pense* qu'elles sont à sens unique.

Une seconde approche peut se baser sur une théorie élémentaire de la complexité. Si l'on suppose que le calcul de $E_k(m)$ prend un temps élémentaire $u$. Il est en théorie simple de calculer *tous* les $E_k(m_i)$ cela prend un temps total card(M)u. La liste des couples $(m, c)$ peut alors être triée par ordre croissant de $c$ en un temps $O(n \ln n)$ par l'un des algorithmes optimaux connus. Le décodage du codon $c$ se fait alors en temps logarithmique par une recherche dichotomique dans la liste. On dira que la fonction est à sens unique si l'on ne connaît pas de meilleur algorithme de calcul de $D_k$ que la méthode qui vient d'être ainsi décrite, soit mieux que $O(\mathrm{card(M)} \ln \mathrm{card(M)})$. Pourtant si cette définition peut sembler plus effective, on ne connaît toujours pas de fonction qui seraient à sens unique selon ce critère, moins restrictif. On essaye donc d'imaginer des fonctions qui pourraient être des fonctions à sens unique en liaison avec des problèmes difficiles.

Parmi les problèmes qui permettent de construire des fonctions à sens unique, deux cas particuliers sont importants : le problème de la factorisation et le problème du logarithme discret. Un autre problème a été considéré dans le passé, le *problème du sac à dos*, mais les tentatives de l'utiliser pour faire de la cryptographie se sont avérées décevantes.

**Le problème de la factorisation** . Étant donnés deux « grands » nombres premiers $p$ et $q$ , il est assez simple de calculer le produit $n = pq$ une multiplication sur des entiers longs. Par contre, étant donné $n$ « grand » il est très difficile de trouver $p$ et $q$ il y a en effet beaucoup d'essais à faire et l'on ne connaît pas, à ce jour, d'algorithme permettant d'en diminuer le nombre.

**Le problème du logarithme discret** . Étant donné un groupe $G =< g >$ « grand » et un entier $k$, il est assez facile de calculer $g^k$, mais étant donné un élément $h \in G$, il est difficile de trouver le logarithme discret de $h$ c'est à dire $k$ tel que $g^k = h$. Là encore il faut faire un nombre élevé d'essais et on ne connaît pas à ce jour d'algorithme efficace permettant d'en diminuer le nombre [1].

Ces deux problèmes permettent de construire des systèmes cryptographiques que nous allons présenter.

1. Il est à noter que les groupes « abstraits »peuvent être conçus comme des groupes de points, de nombres, de fonctions, de matrices, etc. Certaines façons de se représenter un groupe peuvent amener des simplifications au problème du logarithme discret, qui peut devenir sous-exponentiel. Ce n'est pas le cas pour la représentation des groupes présentés ici.

## B.5 Factorisation et système RSA

On se fixe deux « grands » entiers premiers $p \neq q$. Soit $n = pq$. Il est assez facile de calculer $n$ et l'indicateur d'Euler (en anglais *totient*) de $n$, noté $\phi(n)$ et qui est égal par définition au nombre d'entiers inférieurs à $n$ et premiers avec $n$. En effet $\phi(n) = (p-1)(q-1)$. Soit $e$ un entier compris entre 2 et $\phi(n)$ premier avec $\phi(n)$ l'algorithme d'Euclide étendu [152] permet de trouver $u$ et $v$ tels que $ue + v\phi(n) = 1$ c'est à dire que l'entier $u$ est l'inverse de $e$ modulo $\phi(n)$. La recherche des deux grands nombres premiers $p$ et $q$, celle de $e$ peut prendre un peu de temps mais on l'effectue une fois pour toutes. Celle-ci effectuée, il est possible à Alice, qui est en possession de toutes ces informations, de publier le couple $(n, e)$ qui est sa *clé publique* et $(n, u)$ qui est sa *clé privée*. $e$ est appelé *exposant de chiffrement*, *exposant de clé publique* ou tout simplement *exposant public*. $d$ est appelé *exposant de déchiffrement*, *exposant de clé privée* ou plus simplement *exposant privé*.

### B.5.1 Envois chiffrés et discrétion

Si Bob veut envoyer un message à Alice pour qu'elle soit la seule à pouvoir le lire, il découpe son message en une suite de nombres $m_1, m_2, \ldots, m_r$ et lui envoie la suite des codons $c_i = (m_i)^e \ mod \ n$. Pour le déchiffrer, Alice, qui sait que modulo $n$ , $(x^e)^u = x^{eu} = x$ n'a plus qu'à calculer la suite des $(c_i)^u \ mod \ \ n = m_i$ . Rappelons que ces calculs de puissances sont rapides grâce à l'algorithme de l'exponentiation dichotomique (cf. section B.2.1). Quant à Eve qui voudrait bien savoir ce qui a été transmis, même s'il dispose de la suite des codons $c_i$, alors même qu'il connaît $n$ et $e$ qui sont publics, alors même qu'il sait ce qu'il y aurait lieu de faire pour décoder, il ne le peut. En effet trouver $u$ est certes facile si l'on connaît $e$ et $\phi(n)$ mais c'est difficile si l'on connaît seulement $e$ et $n$ car il faut factoriser $n$ pour obtenir $\phi(n)$ et cela est difficile.

### B.5.2 Envois signés et authentification

Si Alice veut envoyer à Bob un message que celui-ci reconnaîtra avec certitude comme authentique, il lui suffit de l'encoder avec sa clé privée. Bob, qui est en possession de la clé publique sera donc a même de le décoder, et comme Alice est seule en possession de sa clé privée, cela convaincra Bob de l'authenticité du message. Pourtant en procédant ainsi, Alice permet à Eve, qui connaît, la clé publique d'Alice comme tout un chacun de prendre connaissance du contenu du message.

Pour obvier à cette difficulté, Bob peut, *lui aussi*, choisir un couple de clé publique/clé privée et divulguer à Alice secrètement sa clé publique, voire la diffuser publiquement comme l'a déjà fait Alice pour la sienne. Alors si l'on note $E_A$ (resp $E_B$) l'encodage d'un message au moyen de la clé publique de $A$ (resp. $B$) et $D_A$ (resp $D_B$) l'encodage d'un message au moyen de la clé privée de $A$ (resp. $B$), Alice peut envoyer à Bob le message $E_B(D_A(M))$. Ce message parvient à Bob qui le considère comme encodé et le décode par $D_B$ qu'il est le seul a connaître. En utilisant alors $E_A$ il obtient le message original d'Alice qu'il sait authentique car seule Alice disposait de $D_A$. Eve aurait bien pu savoir le contenu de l'échange si elle avait été en possession de $D_A(M)$ puisqu'il connaît

la clé publique d'Alice. Mais justement il ne sait obtenir $D_A(M)$ au moyen de $E_B(D_A(M))$.

En fait on peut gagner du temps de calcul en utilisant un *condensat* (en anglais un *hash*). Soit par exemple connue une fonction $H$ définie sur l'ensemble des messages possibles et à valeurs dans l'ensemble des suites de 32 octets. Comme il y a une quantité considérable de messages $M$ possibles et seulement(!) $2^{256}$ valeurs pour H(M), H n'est certainement pas bijective. Un couple $(M', M'')$ de messages tels que $H(M') = H(M'')$ s'appelle une *collision*. On pourrait par exemple imaginer de découper le message en tronçons de 32 octets et de faire la somme, cela nous donnerait une (bien mauvaise) fonction $H$. En fait on recherche des fonctions qui changent de valeurs lorsque l'un des octets du message augmente de 1, changent aussi de valeur lorsque l'on échange deux octets du message, changent de valeurs si l'on fait une permutation circulaire de 3 octets du message, et aussi si l'on échange les parties hautes et basses d'un octet quelconque du message. On dira qu'une fonction de hachage est cryptographique si étant donné un message $M$, trouver un autre message $M'$ tel que $H(M') = H(M)$ est un problème difficile. On connaît des fonctions (par exemple SHA-1 [153]) qui ont ces propriétés. On se donne une telle fonction $H$ et on la rend publique. Pour un message $M$ donné, $H(M)$ sera appelé le condensat de $M$.

Revenant au problème d'Alice de transmettre à Bob un message qu'il saura identifier, elle pourra transmettre $E_B(M+D_A(H(M)))$ où le signe « + » indique la concaténation. Bob (et lui seul) pourra en déduire en appliquant $D_B$ le message M et le « condensat codé » $D_A(H(M))$ il peut alors le décoder au moyen de $E_A$ et vérifier que $H(M) = E_A(D_A(H(M)))$ garantissant l'authenticité. Les encodages supplémentaires on été faits sur des condensats beaucoup plus courts que les messages, et donc au prix de calculs moins coûteux.

## B.6 Logarithme discret et système El-Gammal

On se fixe un groupe $G =< g >$ dont on a une représentation commode. Si $n$ est son ordre, on a $G = \{e, g, g^2, \ldots, g^{n-1}\}$. En pratique, on choisit $n$ très grand, par exemple $n > 10^{50}$. Autant dire qu'il est hors de question de stocker ce groupe dans une mémoire d'ordinateur [2]. C'est à dire que les différents éléments du groupe ne pourront individuellement qu'être *calculés*, d'où l'importance d'une bonne représentation. Individuellement, ces éléments de groupe pourront être facilement, rangés ou échangés car on peut les stocker sur environ $\ln(10^{50})/\ln(2) = 166$ bits. Les éléments du code $G$, $g$ et $n$ sont rendus publics.

Si Alice veut publier une clé publique elle choisit au hasard un élément $u \in \{1, 2, \ldots, n-1\}$, calcule $k_P = g^u$, elle garde secret l'entier $u$ et publie $k_P$ .

Maintenant lorsque Bob veut envoyer en toute sécurité un message à Alice il le représente par un entier qui correspond à un élément $m \in G$, il choisit au hasard un élément $v \in \{1, 2, \ldots, n-1\}$ et calcule une seule fois $k'_S = k_P^v$ , il calcule enfin $g^v$ et $m \cdot k'_S$. Il envoie à Alice le message $(g^v, m \cdot k'_S)$.

2. En utilisant la masse atomique du silicium($m_{\mathrm{Si}} = 28$), de l'oxygène($m_{\mathrm{O}} = 16$) et le nombre d'Avogadro ($N = 6.02\ 10^{23}$). Si l'on utilisait un atome de silicium pour stocker chaque élément du groupe, et l'on sait qu'il en faut beaucoup plus, on peut calculer ainsi la masse de sable (c'est à dire de silice $SiO_2$) qu'il faudrait pour extraire le silicium nécessaire. Il faudrait en effet tout le silicium contenu dans $((28 + 2 \times 16) \times 10^{50})/(1000 \times N)$ kilogrammes de sable, au total $9,9610^{24}$ kilogrammes de sable, un peu plus que la masse de la terre !

Alice peut alors calculer rapidement $k_S = (g^v)^u$, puisqu'elle a reçu $g^v$ et qu'elle connaît $u$. Elle peut alors calculer $k_S = (k'_S)^{-1} = (g^v)^{n-u} = (g^n)^v \cdot g^{-uv}$ car $(g^n)^v = e^v = e$. Puis elle peut calculer $(m \cdot k'_S) \cdot k_S = m$

Maintenant Eve, qui écoute l'échange et qui voudrait bien savoir ce qui s'est dit, ne le peut pas : elle connaît certes $g^u$ qui a été publié mais ne peut pas en déduire $u$ car c'est un calcul de logarithme discret bien trop long à effectuer. De même, elle connaît aussi $g^v$ mais ne peut trouver $v$. Il ne lui est donc pas possible de retrouver $k'_S$ et donc $k_S$, et cela la met dans l'impossibilité de retrouver $m$.

Remarquons qu'alors que $k_P$ est fixée, $k_S$ dépend du choix de Bob et peut donc changer à chaque nouvel échange. On parle pour cela d'une *clé de session*.

Les mécanismes de signature évoqués à la section précédente s'adaptent immédiatement, *mutatis mutandis* à cette méthode de chiffrement. Nous ne les décrirons pas à nouveau.

S'il a été décrit initialement pour un groupe de la forme $\mathbb{Z}_p$, le système s'applique immédiatement à *n'importe quel* groupe cyclique, par exemple ceux obtenus à partir d'une courbe elliptique (cf. section B.2.2), à partir de la variété de Picard [154] d'une courbe hyperelliptique, etc.

Malgré ses attraits le système El Gamal a un inconvénient : le message chiffré échangé est d'une taille approximativement double de celle du message initial. Pour cette raison, on l'utilise plutôt dans des *systèmes hybrides* où il permet d'échanger une clé symétrique qui sera utilisée par la suite pour les échanges.

## B.7 D'autres systèmes

On se rend compte très vite que de nombreux systèmes peuvent être envisagés sur les principes semblables à ceux que l'on a étudiés plus-haut, pour peu que l'on connaisse des groupes cycliques dont les éléments peuvent être représentés commodément. Nous avons vu que les courbes elliptiques permettent un tel travail. L'implantation des algorithmes relatifs aux courbes elliptiques a déjà été faite dans la bibliothèque publique OpenSSL. Des études expérimentales intenses ont permis de caractériser *certaines* courbes elliptiques particulières donnant des systèmes de cryptographie de bonne qualité. Ces courbes sont répertoriées et leurs noms codés. Plus généralement les courbes *hyperelliptiques*, qui ne sont pas naturellement munies d'une structure de groupe, ne permettent pas néanmoins la construction de systèmes cryptographiques en considérant la variété de Picard [154] associée à ces courbes. Enfin au lieu de considérer le groupe des inversibles de $\mathbb{Z}_p$ où $p$ est grand, on peut considérer, si $q = p^k$, le groupe des inversibles de $\mathbb{F}_q$ dont les éléments se représentent naturellement comme des combinaisons linéaires formelles d'éléments de $Z_p$ avec de bonnes propriétés de calcul. Cela donne le système XTR [155].

L'important reste pourtant une communauté de principes de conception qui permet de construire pour chacun de ces systèmes cryptographiques un système de cryptographie à seuil associé. N'importe lequel de ceux-ci peut être mis en place dans le réalisation des protocoles.

# Annexe C

# SEcure Neighbor Discovery (SEND)

SEND est une extension du protocole de découverte de voisins (en anglais *Neighbor Discovery Protocol* ou NDP), un protocole qui joue un rôle important dans les réseaux utilisant le protocole IPv6. Afin de comprendre son mode d'opération, nous présentons dans une première section, les principaux aspects du protocole IPv6 et en particulier le mécanisme d'auto-configuration d'adresses, ensuite, dans une deuxième section, les principales caractéristiques du protocole NDP et tout particulièrement le mécanisme de détection d'adresses dupliquées. Les mécanismes d'auto-configuration d'adresses et de détection d'adresse dupliquées vont de pair afin de s'assurer de l'unicité des adresses IP dans un réseau. La dernière section traite du fonctionnement du protocole SEND et des adresses générées de façon cryptographique (*Cryptographically Generated Addresses* ou CGAs) qui sont au cœur de ce protocole.

L'ensemble de ces mécanismes a fait l'objet de standards à l'IETF et est destiné à fonctionner dans les réseaux à infrastructure. L'adaptation des mécanismes d'auto-configuration d'adresse et de détection d'adresses dupliquées au contexte des MANETs est en cours d'étude au sein des groupes MANET [4] et Autoconf (*Ad Hoc Network Autoconfiguration*) [73] de l'IETF. Des propositions existent aussi sous forme d'articles. N'étant pas au cœur de cette thèse, nous en donnerons brièvement une idée sans les détailler dans les sections correspondantes.

## C.1 Internet Protocol version 6 (IPv6)

Le protocole IPv6, défini dans le RFC 2460 [156], précise un en-tête pour les paquets, au niveau 3 de la pile de protocoles. Outre les champs précisant la version du protocole IP, la longueur des données transportées par le paquet, la classe du trafic, l'en-tête suivant [1] et plusieurs autres informations, cet en-tête inclut les adresses de la source et de la destination du paquet. Par métonymie, on appelle de telles adresses *adresses IPv6*.

1. L'en-tête suivant contient un entier permettant d'identifier le protocole du message contenu dans le paquet IP. Trois possibilités existent : ICMPv6 pour les messages de contrôle,

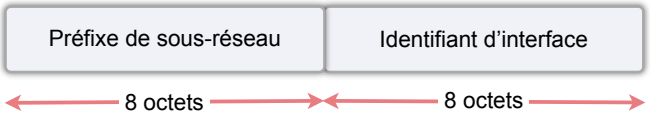


FIGURE C.1 – Adresse IPv6

Une adresses IPv6 est de longueur 128 bits, soit 16 octets, et, comme les adresses IPv4, est constituée de deux parties (cf. FIGURE C.1) :

- préfixe de sous-réseau : il est formé par les 64 bits, soit 8 octets, de poids fort de l'adresse et est commun à tous les équipement se trouvant dans le même sous-réseau
- identifiant d'interface : il est formé par les 64 bits de poids faible de l'adresse et peut être calculé de différentes façons ou obtenu d'un serveur, par exemple un serveur DHCP

Une adresse IPv6 peut soit être configurée manuellement dans une interface réseau d'un équipement, soit être auto-configurée.

**Auto-configuration d'adresses IPv6** Il existe deux procédés d'auto-configuration : l'auto-configuration avec état et l'auto-configuration sans état. Dans le premier cas, un serveur DHCPv6 (*Dynamic Host Configuration Protocol* version 6) envoie une adresse à l'équipement [2], tandis que dans le deuxième cas, l'équipement définit, de façon autonome, son adresse IPv6.

Lors d'une configuration sans état, l'équipement considéré commence par calculer un identifiant d'interface, par exemple en se servant de l'adresse de niveau 2 [157] c.-à-d. de la couche liaison de données, appelée aussi adresse MAC, ou, aussi, en utilisant un certain nombre de paramètres, appelés paramètres CGA, afin d'obtenir une adresse du même nom (cf. section C.3.1). Il concatène, ensuite, cet identifiant au préfixe de lien local [3] défini par le RFC 2460 [156] pour obtenir une adresse de lien local.

Avant de pouvoir configurer cette adresse à l'une de ses interfaces, l'équipement doit d'abord vérifier qu'elle n'est pas en cours d'utilisation par un autre nœud se trouvant sur le même lien que lui. Pour cela, il exécute le mécanisme de détection d'adresses dupliquées (en anglais *Duplicate Address Detection* ou DAD) qui fait partie intégrante des fonctionnalités du protocole NDP (cf. section C.3). Si l'adresse se révèle être unique, elle est finalement configurée dans l'interface considérée. Sinon, un nouveau identifiant d'interface est calculé et l'unicité de l'adresse obtenue est testée.

---

UDP ou TCP pour le transport des données.

2. DHCPv6 utilise le modèle client/serveur c.-à-d. que l'équipement, souhaitant acquérir une adresse IP, doit contenir un client DHCPv6 capable de dialoguer avec un serveur DHCPv6 pour demander une telle adresse. Le serveur DHCPv6 délivre non seulement l'adresse IPv6, mais aussi, selon sa configuration, d'autres informations comme la durée de validité de cette adresse, la passerelle par défaut à utiliser, le serveur DNS, etc.

3. Un lien est un moyen physique par lequel les équipements peuvent communiquer au niveau 2 avec des protocoles comme Ethernet, PPP ou X.25 sans passer par un équipement de routage. Au sein de ce lien, une adresse de lien local a une portée limitée c.-à-d. que seuls les équipements voisins, qui sont attachés à ce lien, peuvent recevoir les paquets émis par l'interface utilisant cette adresse. Elle n'est donc interprétable que par les équipements du voisinage.

Afin que son adresse soit utilisable au delà du lien local et que ses paquets soient transmis par un routeur, un nœud est amené, par la suite, à récupérer le préfixe du réseau auquel il appartient en écoutant ou en sollicitant des messages NDP de type *annonce de routeur* envoyés par les routeurs qui se trouvent sur le même lien que lui. Ce préfixe remplacera le préfixe de lien local dans l'adresse.

Dans un réseau MANET, au moment de l'auto-configuration de son adresse, un nœud peut ne pas se trouver à la porter d'un routeur et, ainsi, ne pas recevoir des *annonces de routeur*. Le travail effectué dans [158] préconise d'utiliser alors le préfixe de sous-réseau par défaut, MANET_PREFIX, dont la valeur est fec0:0:0:ffff::/64. Une autre solution propose plutôt d'étendre le mécanisme d'*annonce de routeur* au delà du voisinage du routeur de l'opérateur, s'il existe, en relayant les messages NDP [159].

## C.2 Neighbor Discovery Protocol (NDP)

Le protocole de découverte de voisins, défini par le RFC 2461 [160], utilise des messages ICMPv6 (*Internet Control Message Protocol* version 6) qui sont directement encapsulés dans des messages IPv6 (cf. FIGURE C.2). En plus de

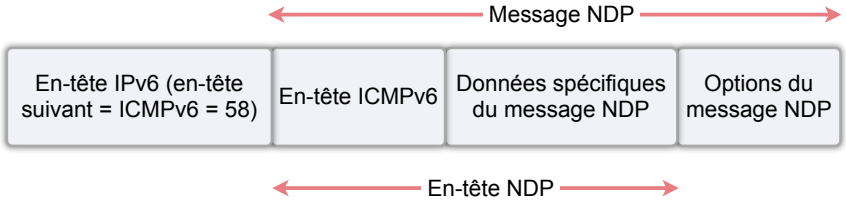


FIGURE C.2 – Format d'un message NDP

l'en-tête NDP, qui consiste en un en-tête ICMPv6 [4] suivi d'un ensemble de données spécifiques à NDP, un message NDP contient une ou plusieurs options. Une option du message NDP est formée d'un champ type et d'un champ longueur suivis d'autres champs dont le contenu dépend du type de l'option. Parmi ces types, on peut citer l'adresse de couche 2 utilisée par exemple dans la résolution d'adresses, l'information sur un préfixe utilisée dans la découverte du préfixe du réseau et l'en-tête de redirection utilisé dans la redirection[160].

Un message NDP peut avoir l'un des cinq types suivants :

- une sollicitation de routeur (en anglais *Router Solicitation* ou RS),
- une annonce de routeur (en anglais *Router Advertisement* ou RA), envoyée en réponse à un message RS émis par un nœud se trouvant sur le même lien. Elle peut être aussi envoyée en l'absence de toute sollicitation,
- une sollicitation de voisin (en anglais *Neighbor Solicitation* ou NS),
- une annonce de voisin (en anglais *Neighbor Advertisement* ou NA), envoyée suite à un message RA émis par un nœud voisin se trouvant sur le même lien (cf. FIGURE C.3). Elle peut être aussi envoyée en l'absence de toute sollicitation,

4. L'en-tête ICMPv6 est constitué par les champs *type*, *code* qui vaut toujours 0 et *checksum*.

– une redirection (en anglais *Redirect*).

Ces différents types de messages, selon leur contenu en données spécifiques et en options, implémentent tout un panel de fonctions allant de la résolution d'adresses à la détection d'inaccessibilité de voisins, la découverte de routeurs et de voisins, la détection d'adresses dupliquées (DAD), la redirection, etc [160].

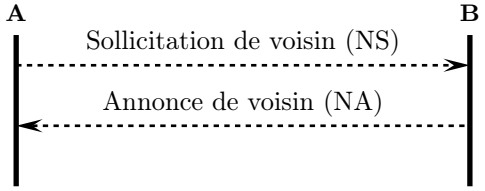


FIGURE C.3 – Dialogue NDP entre nœuds voisins

Le mécanisme de détection d'adresses dupliquées est décrit dans le RFC 2462 [157] et exploite les deux messages NDP : NS et NA. Avant de configurer une adresse IPv6 dans l'une de ses interfaces de réseau, un nœud procède à la vérification de son unicité dans le voisinage. Pour cela, il envoie un message NS contenant dans son champ *adresse de la cible* l'adresse à vérifier. S'il existe un autre nœud du réseau qui l'utilise déjà, on dit qu'une *collision d'adresses* a été détectée et le nœud récepteur doit répondre au nœud initiateur par un message NA (cf. FIGURE C.3). Plusieurs cas de figure doivent, par ailleurs, être pris en compte lors de l'exécution du DAD [157].

Par ailleurs, le groupe Autoconf à l'IETF [73] travaille à la définition des mécanismes d'auto-configuration d'adresses IP par les nœuds ad hoc. Dans ce cadre une extension du DAD, DAD++, a été proposée dans [161] afin de détecter les collisions d'adresses au delà du voisinage directe. Pour cela, un nœud procède en deux temps : il envoie d'abord un message NS à ses voisins et, si aucune collision d'adresses n'a été signalée par un message NA envoyé par l'un d'entre eux, un autre message NS est alors diffusé dans le réseau MANET. La nature de ce message et les nouvelles options qu'il contient sont décrits dans [161]. Il est à noter que l'adresse ne doit pas contenir un préfixe de lien local afin que le message NS soit relayé.

## C.3 SEND

SEND a été conçu pour résoudre les problèmes de sécurité qui menacent le protocole NDP [162] lorsque la sécurité du lien physique n'est pas assurée. Grâce à SEND, NDP devient utilisable dans ce type d' environnement comme dans les réseaux sans-fil où les attaques de NDP sont possibles. Ce protocole opère par le biais de nouvelles options NDP définies dans le RFC 3971 [70] et se base sur les CGAs. Les options sont au nombre de quatre et sont : *l'option CGA*, *l'option RSA*, *l'option horodatage* (en anglais *Timestamp*) et *l'option Nonce*.

Un nœud supportant SEND doit être muni d'un couple de clés publique et privée afin de pouvoir calculer aussi bien son adresse CGA que l'option RSA à inclure dans ses messages SEND. Ci-dessous, nous présentons brièvement les algorithmes de génération et de vérification de CGAs ainsi que les différentes options de SEND. Il est à noter que tout message SEND doit avoir comme

adresse IPv6 source l'adresse CGA de l'émetteur. Il doit, en plus, contenir toutes les options décrites à la section C.3.2.

### C.3.1 Cryptographically Generated Address (CGA)

Une adresse générée cryptographiquement (*Cryptographically Generated Address* ou CGA) [74] est une adresse IPv6 dont l'identifiant d'interface est lié de façon cryptographique à la clé publique du nœud qui l'exploite. L'objectif de cette technique est d'éviter l'attaque par vol d'adresse. Son emploi est, pour cette raison, indissociable de la signature électronique. En effet, il ne suffit pas, pour garantir qu'une telle adresse appartient bien à une entité donnée, de vérifier qu'elle a été calculée en employant sa clé publique. Il faut, en plus, vérifier que la clé publique lui appartient bien.

La génération d'une CGA par un nœud source et la vérification de sa validité par un nœud destinataire font appel à un ensemble de données, appelées *paramètres CGA*. Leurs valeurs sont susceptibles de changer au cours de l'exécution de l'algorithme de génération de CGA. En revanche, ces valeurs deviennent définitives une fois que l'absence de collision de la CGA avec les autres adresses IPv6 dans le réseau, par applications successives du DAD (cf. section C.3), a été établie. Elles sont alors figées dans une *structure de données de paramètres CGA* au sein de l'option CGA (cf. section C.3.2).

Une structure de données de paramètres CGA comporte les éléments suivants :

- un compteur de collision : un entier incrémenté lorsqu'après exécution du DAD l'adresse testée s'est révélée appartenir à une autre entité. Sa valeur commence à 0 et ne doit pas excéder 2,
- un modificateur (en anglais *modifier*) : un nombre aléatoire dont la valeur a pu changer au cours de l'exécution de l'algorithme de génération de CGA,
- le préfixe de sous-réseau auquel appartient l'entité considérée,
- la clé publique de l'entité considérée : une clé RSA de longueur allant de 384 à 2048 bits,
- des champs supplémentaires réservés à d'éventuelles extensions.

Le message SEND était transporté dans un paquet comportant la CGA en tant qu'adresse source. Il contient dans son corps l'option CGA. Elles sont toutes les deux nécessaires à chaque nœud récepteur désireux de vérifier, au moins en partie, la validité de la CGA, c.-à-d. sa réelle appartenance au nœud émetteur.

L'algorithme de vérification de validité d'une CGA se déroule selon les étapes suivantes :

1. vérifier que la valeur du compteur de collision est 0, 1 ou 2. Dans le cas contraire, la CGA n'est pas valide.
2. vérifier que le préfixe de sous-réseau figurant parmi les paramètres CGA est égal à celui de l'adresse IPv6 source figurant dans l'en-tête IPv6, qui est la CGA à vérifier.
3. calculer un identifiant d'interface au moyen des paramètres CGA obtenus et comparer sa valeur à celle de l'identifiant d'interface de la CGA reçue. Si certains bits ne sont pas identiques, à part quelques uns bien identifiés dans le RFC 3972 [74], la vérification de la CGA échoue.

4. faire des vérifications supplémentaires nécessaires à la validité de la CGA en calculant d'autres éléments au moyen de la clé publique et du modificateur.

Lorsque toutes les étapes réussissent, l'appartenance de la CGA au nœud possédant la clé publique se trouvant dans l'option CGA est établie. Il reste alors à vérifier la validité de la signature RSA figurant dans l'option RSA (cf. C.3.2). Lorsque toutes ces vérifications réussissent, un nœud récepteur crée une entrée dans son cache, appelé *cache de voisins* (en anglais *Neighbor cache*), pour y enregistrer les paramètres CGA et l'adresse CGA de son nouveau voisin dont il vient de recevoir un message SEND. Ainsi, en cas de besoin, il lui sera possible ultérieurement de vérifier à nouveau la validité de cette adresse, par exemple s'il reçoit d'autres messages de ce voisin, sans avoir, pour autant, besoin de se faire envoyer à nouveau les paramètres CGA. Ce cache est aussi un moyen de garder une trace de tous les voisins connus.

### C.3.2 Options SEND

**Option CGA** L'option CGA contient les paramètres CGA permettant à un nœud destinataire d'un message SEND de vérifier la validité de l'adresse IPv6 source du paquet c.-à-d. de vérifier qu'elle n'a pas été volé et qu'elle appartient bien au nœud émetteur.

Les nœuds supportant SEND doivent ajouter l'option CGA à tous leurs messages de sollicitation et d'annonce de voisins et aux messages de sollicitation de routeur.

**Option d'horodatage** L'option d'horodatage contient un entier représentant le nombre de secondes écoulées depuis le 1er janvier 1970 00 :00 :00 UTC [5] jusqu'à l'événement à dater, à savoir l'instant de génération du message SEND. Elle est incluse dans tout type de message SEND. Son but est d'assurer qu'un message de redirection ou un message d'annonce de voisins ou de routeurs qui n'a pas été sollicité ne soit pas envoyé, à nouveau, par un nœud malveillant et que ce dernier ne soit pas en mesure de mener, ainsi, une attaque par rejeu (cf. section 2.4.3.5 du chapitre 2).

**Option Nonce** L'option Nonce contient un nombre aléatoire généré par le nœud émetteur du message SEND. Elle est utilisée dans des messages de sollicitation et d'annonce afin de garantir qu'un message d'annonce est bien une réponse, fraîchement envoyée, à un message de sollicitation. Pour cela, le message d'annonce doit contenir la valeur du nonce du message de sollicitation qui l'a déclenché.

**Option RSA** L'option RSA contient, pour l'essentiel, un haché de la clé publique du nœud émettant le message SEND et une signature sur un groupe de données. La clé publique doit être associée à la clé privée qui a servi à générer cette signature. Le calcul de cette dernière fait appel à l'algorithme de signature

5. L'UTC ou *Temps Universel Coordonné* est, selon l'observatoire de Paris : une « échelle de temps, maintenue par le BIPM (Bureau International des Poids et Mesures) et le service international de la rotation terrestre (IERS), qui constitue la base d'une diffusion coordonnée, des fréquences étalon et des signaux horaires ».

de RSA et à l'algorithme de hachage SHA-1 et porte sur le groupe de données suivant :

- l'étiquette du type de message CGA,
- les adresses IPv6 source et destination du message SEND se trouvant dans l'en-tête IPv6,
- les champs type, code et checksum de l'en-tête ICMPv6,
- le reste de l'entête du message SEND, partant du champ suivant le checksum jusqu'au champ avant les options,
- toutes les options incluses dans le message SEND et venant avant l'option RSA. Comme un nœud doit d'abord ajouter toutes les options à son message SEND, sauf l'option RSA qu'il ajoute à la fin, la signature porte sur l'ensemble des options.

L'option RSA doit être ajoutée aux messages SEND de sollicitation et d'annonce de voisins et à certains messages de sollicitation de routeurs. Un routeur doit l'ajouter à ses messages d'annonce de routeur et de redirection. A la réception d'un message SEND contenant une option RSA, un nœud doit la valider en vérifiant, en particulier, que le haché de la clé correspond à une clé publique connue auparavant, par exemple grâce à celle figurant dans l'option CGA. Il doit aussi vérifier que la signature est valide c.-à-d. que sa construction a bien suivi les recommandations du PKCS#1 v1.5 [163].

# Annexe D

# Mécanisme d'accès de base de la fonction DCF

Ainsi que cela est spécifié dans le standard IEEE 802.11 [26], la fonction *Distributed Coordination Function* (DCF), un mécanisme fondamental à l'accès au canal, est basée sur le protocole *Carrier Sense Multiple Access with Collision Avoidance* ou CSMA/CA. Elle décrit deux techniques possibles de transmission de paquets. Le mécanisme d'accès de base (en anglais *basic access mechanism*) est la technique par défaut. Elle consiste en une poignée de main soit un message aller et un message retour. La deuxième technique utilise des messages RTS/CTS (*Request-to-send/Clear-to-send*) et consiste en un protocole à quatre messages, soit deux aller-retours. Nous présentons dans cette annexe la technique d'accès de base telle que résumée dans [164].

Avant de transmettre un nouveau paquet, un nœud observe et suit l'activité dans le canal. Si le canal reste libre pendant une durée égale à un DIFS (*Distributed Inter-Frame Space*), le nœud transmet. Sinon, si une activité est détectée et que le canal est occupé, immédiatement ou durant le DIFS, le nœud continue à observer le canal jusqu'à ce que celui-ci devienne libre pendant un DIFS. Dans ce cas, le nœud calcule une durée aléatoire de temporisation (en anglais *random backoff interval*) pendant laquelle il attend avant de transmettre afin de minimiser la probabilité de collision avec des paquets transmis par d'autres nœuds [1]. De plus, afin d'éviter de monopoliser le canal, un nœud doit attendre un temps aléatoire entre deux transmissions consécutives de deux paquets différents même si le medium a été détecté libre durant l'intervalle DIFS.

Pour des raisons d'efficacité, le mécanisme DCF emploie une échelle de temporisation à temps discret (en anglais *discrete-time backoff scale*). Le temps qui suit immédiatement un intervalle DIFS, où le canal est resté libre, est échelonné. Il est permis à un nœud de transmettre seulement au début de chaque *slot* de temps. La taille $\theta$ d'un *slot* de temps dépend de la couche physique et est égale au temps nécessaire à un nœud pour détecter la transmission d'un paquet provenant d'un autre nœud.

Le mécanisme DCF adopte un schéma de temporisation exponentiel (en anglais *exponential backoff scheme*). Avant la transmission de chaque paquet,

1. En termes plus imagés, on évite ainsi que tous ceux qui attendent se précipitent sur le premier *slot* libre provoquant, par là même, une rafales de collisions.

| PHY | *slot* de temps ($\theta$) | $CW_{min}$ | $CW_{max}$ |
|---|---|---|---|
| FHSS | 50 $\mu s$ | 16 | 1024 |
| DSSS | 20 $\mu s$ | 32 | 1024 |
| IR | 8 $\mu s$ | 64 | 1024 |

TABLEAU D.1 – Valeurs des fenêtres de contention pour les trois types de couche physique spécifiées par le standard IEEE 802.11 : FHSS (Frequency-Hopping Spread Spectrum), DSSS (Direct Sequence Spread Spectrum), et IR (Infra-Red)

un nombre aléatoire $r$ est choisi uniformément dans la plage $\{0, \cdots, cw - 1\}$ et le nœud attend pendant le temps de temporisation $r\theta$. $cw$ est nommée *fenêtre de contention* et sa valeur dépend du nombre de transmissions échouées pour un paquet donné. A la première tentative de transmission, la valeur de $cw$ est fixée au minimum de la fenêtre de contention c.-à-d. $CW_{min}$. Après chaque transmission non réussie, la valeur de $cw$ est doublée jusqu'à la valeur maximum $CW_{max} = 2^m\, CW_{min}$. Les valeurs de $CW_{min}$ et $CW_{max}$ dépendent du type de la couche physique. Elles sont résumées dans le tableau D.1.

Le compteur du temps de temporisation (en anglais *backoff time counter*) est décrémenté tant que le canal est détecté comme libre, *gelé* quand une transmission est détectée et réactivé lorsque le canal redevient libre pendant un intervalle supérieur à DIFS. Le nœud transmet lorsque le compteur du temps de temporisation atteint zéro.

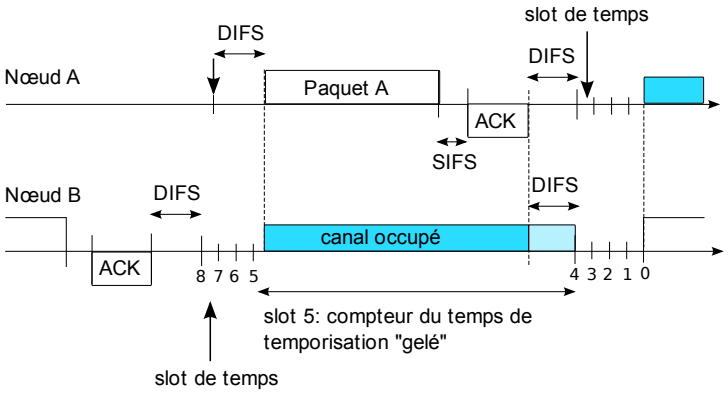


FIGURE D.1 – Exemple illustrant le mécanisme d'accès de base

Le mécanisme CSMA/CA ne repose pas sur la capacité des nœuds à détecter une collision en écoutant leurs propres transmissions. Pour cette raison, un message d'acquittement, noté ACK, est envoyé par le nœud destinataire afin de notifier au nœud source la réussite de la réception. Le message ACK est envoyé immédiatement à la fin de la réception du paquet de la source et après attente pendant un intervalle de temps appelé SIFS (*Short Inter-Frame Space*). Comme l'intervalle SIFS est plus court que l'intervalle DIFS, aucun autre nœud

ne pourra détecter le canal comme libre jusqu'à la fin de transmission du message ACK. Si le nœud source ne reçoit pas le message ACK durant un intervalle de temps fixe appelé ACK_Timeout ou s'il détecte la transmission d'un autre paquet sur le canal, il programme à nouveau la transmission de son paquet.

La figure D.1 illustre, à travers un exemple tiré de [164], le fonctionnement du mécanisme d'accès de base. Deux nœuds A et B partagent le même canal sans-fil. A la fin de la transmission de son paquet, le nœud B attend durant un intervalle DIFS ensuite choisit un temps de temporisation égal à 8 avant de transmettre le paquet suivant. On suppose que le premier paquet du nœud A arrive à l'instant indiqué avec une flèche sur la figure. Après l'attente d'un intervalle de temps de longueur DIFS, le paquet est transmis. On note que la transmission du paquet de A a lieu au milieu du *slot* correspondant à la valeur de temporisation 5 pour le nœud B. Le canal est donc détecté comme occupé par B. En conséquence, le temps de temporisation est gelé à sa valeur 5. Le compteur du temps de temporisation est décrémenté à nouveau seulement lorsque le canal, pendant un intervalle DIFS, est détecté à nouveau comme libre.

# Annexe E

# Calculs avec Maple

Dans cette annexe, le calcul du chapitre 6 portant sur le *modèle max* et le *modèle somme* est détaillé. Une feuille de calcul par modèle a été créée et collée ci-dessous dans la section correspondante.

Dans une feuille de calcul, les formules présentées en rouge et précédées du signe supérieur sont celles que nous avons écrites en Maple. Afin d'expliquer le raisonnement qui a été élaboré, elles sont précédées, à chaque fois, d'un commentaire commençant par le signe dièse. Chaque formule est, ensuite, suivie d'un développement du calcul présenté en bleu. Un développement comporte un numéro à la fin de sa première ligne permettant de le repérer dans la feuille de calcul.

Les deux feuilles comportent une partie commune qui s'étend de la première à la huitième formule. Elles peuvent être repérées grâce aux numéros de leurs développements dans la première feuille. Dans le chapitre 6, cela correspond au calcul jusqu'aux équations 6.17 et 6.20. Ainsi, dans la seconde feuille correspondant au *modèle somme*, les formules sont reprises à partir de la neuvième formule.

## E.1 Modèle max

```
> #les messages ACK sont pris en consideration (c.f Annexe E)
  #n      : nbr de pairs AAA
  #hops   : nbr de sauts entre JN et un pair AAA
  #lambda : débit
  #l, l1, l2, l3, l4 : longueurs des messages en octets
  #i      : nbr de retransmission
  #p      : probabilité de retransmission
  #DIFS, SIFS, ACK_Timeout : temporisateurs en secondes
  #CWmin  : fenêtre de contention minimale en secondes
  #ED_ACK : temps d'émission du message ACK
> restart;
> assume(n, posint, lambda>0, l>0, l1>0, l2>0, l3>0, l4>0, i, natural, DIFS, realcons, CWmin,
  realcons, s_i>0, vTRd_k>0, sROTd>0, hops, posint, t, real, p, RealRange(0,1), SIFS, realcons,
  ACK_Timeout, realcons, Ed_ACK, realcons);
> #sur un lien à un seul saut:
  #temps de transfert conditionné au nombre de retransmissions i, son écart type :
  s_i:=sqrt((i+1)*l)/lambda;
```

$$s_i := \frac{\sqrt{(i+1)\,l}}{\lambda} \quad (1$$

```
> #sa moyenne :
  mu_i:=(i+1)*DIFS+(i+1)*l/lambda+1/2*sum(2^j*CWmin,j=0..(i-1))+i*ACK_Timeout+SIFS+Ed_ACK;
```

$$mu_i := (i\sim+1)\,DIFS\sim + \frac{(i\sim+1)\,l\sim}{\lambda\sim} + \frac{2^{i\sim}\,CWmin\sim}{2} - \frac{CWmin\sim}{2} + i\sim ACK_Timeout\sim + SIFS\sim + Ed_ACK\sim \quad (2$$

```
> #il suit une loi normale, sa densité de probabilité :
  fTRd_i:=unapply((1/(s_i*sqrt(2*Pi)))*exp(-(x-mu_i)^2/(2*s_i^2)),x);
```

$$fTRd_i := x \rightarrow \frac{1}{2}\,\frac{\lambda\sim\sqrt{2}\,e^{-\frac{1}{2}\frac{\left(x-(i\sim+1)\,DIFS\sim-\frac{(i\sim+1)\,l\sim}{\lambda\sim}-\frac{2^{i\sim}\,CWmin\sim}{2}+\frac{CWmin\sim}{2}-i\sim ACK_Timeout\sim-SIFS\sim-Ed_ACK\sim\right)^2\lambda\sim^2}{(i\sim+1)\,l\sim}}}{\sqrt{(i\sim+1)\,l\sim}\,\sqrt{\pi}} \quad (3$$

```
> int(fTRd_i(x),x=-infinity..infinity);
```

$$1 \quad (4$$

```
> #sa fonction de répartition :
  FTRd_i:=int(fTRd_i(x),x=-infinity..t);
```

$$FTRd_i := \frac{1}{2}\,\frac{1}{\sqrt{(i\sim+1)\,l\sim}\,\sqrt{\frac{1}{(i\sim+1)\,l\sim}}}\left(-\mathrm{erf}\left(\frac{1}{4\,(i\sim+1)\,l\sim\sqrt{\frac{1}{(i\sim+1)\,l\sim}}}\left(\left(2\,i\sim l\sim+2\,\lambda\sim i\sim ACK_Timeout\sim+2\,\lambda\sim DIFS\sim i\sim\right.\right.\right.\right. \tag{5}$$

$$\left.\left.\left.+2\,\lambda\sim Ed_ACK\sim-2\,\lambda\sim t\sim+2\,\lambda\sim SIFS\sim-\lambda\sim CWmin\sim+2\,l\sim+2\,\lambda\sim DIFS\sim+\lambda\sim 2^{i\sim}\,CWmin\sim\right)\sqrt{2}\right)\right)$$

$$\left.+\sqrt{(i\sim+1)\,l\sim}\,\sqrt{\frac{1}{(i\sim+1)\,l\sim}}\right)$$

```
> #temps de transfert tenant compte de toutes les possibilités de retarnsmission, sa fonction de
  répartition :
  FTRd_k:=collect(simplify(sum((     1/2*(-erf(1/4*(2*i*l+2*lambda*i*ACK_Timeout+2*lambda*DIFS*i+2*
  lambda*Ed_ACK-2*lambda*t+2*lambda*SIFS-lambda*CWmin+2*l+2*lambda*DIFS+lambda*2^i*CWmin)*2^(1/2)/
  (1/((i+1)*l))^(1/2)/(i+1)/l)+((i+1)*l)^(1/2)*(1/((i+1)*l))^(1/2))/((i+1)*l)^(1/2)/(1/((i+1)*l))^
  (1/2)        )*p^i*(1-p),i=0..6)+sum((         1/2*(-erf(1/4*(2*i*l+2*lambda*i*ACK_Timeout+2*
  lambda*DIFS*i+2*lambda*Ed_ACK-2*lambda*t+2*lambda*SIFS-lambda*CWmin+2*l+2*lambda*DIFS+lambda*2^i*
  CWmin)*2^(1/2)/(1/((i+1)*l))^(1/2)/(i+1)/l)+((i+1)*l)^(1/2)*(1/((i+1)*l))^(1/2))/((i+1)*l)^(1/2)/
  (1/((i+1)*l))^(1/2)        )*p^i,i=7..7)),p);
```

$$FTRd_k := \left(-\frac{1}{2}\,\mathrm{erf}\left(\frac{16\,l\sim+14\,\lambda\sim ACK_Timeout\sim+16\,\lambda\sim DIFS\sim+2\,\lambda\sim Ed_ACK\sim-2\,\lambda\sim t\sim+2\,\lambda\sim SIFS\sim+127\,\lambda\sim CWmin\sim}{8\sqrt{l\sim}}\right)\right. \tag{6}$$

$$\left.+\frac{1}{2}\,\mathrm{erf}\left(\frac{\left(14\,l\sim+12\,\lambda\sim ACK_Timeout\sim+14\,\lambda\sim DIFS\sim+2\,\lambda\sim Ed_ACK\sim-2\,\lambda\sim t\sim+2\,\lambda\sim SIFS\sim+63\,\lambda\sim CWmin\sim\right)\sqrt{2}\,\sqrt{7}}{28\sqrt{l\sim}}\right)\right)$$

$$p\sim^{7}+\left(\right.$$

$$\left.-\frac{1}{2}\,\mathrm{erf}\left(\frac{\left(14\,l\sim+12\,\lambda\sim ACK_Timeout\sim+14\,\lambda\sim DIFS\sim+2\,\lambda\sim Ed_ACK\sim-2\,\lambda\sim t\sim+2\,\lambda\sim SIFS\sim+63\,\lambda\sim CWmin\sim\right)\sqrt{2}\,\sqrt{7}}{28\sqrt{l\sim}}\right)\right.$$

$$+ \frac{1}{2} \operatorname{erf}\left( \frac{(12\, l\sim + 10\, \lambda\sim ACK_Timeout\sim + 12\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 31\, \lambda\sim CWmin\sim)\sqrt{3}}{12\sqrt{l\sim}} \right) \Bigg) p\sim^{6}$$

$$+ \Bigg( - \frac{1}{2} \operatorname{erf}\left( \frac{(12\, l\sim + 10\, \lambda\sim ACK_Timeout\sim + 12\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 31\, \lambda\sim CWmin\sim)\sqrt{3}}{12\sqrt{l\sim}} \right)$$

$$+ \frac{1}{2} \operatorname{erf}\left( \frac{(10\, l\sim + 8\, \lambda\sim ACK_Timeout\sim + 10\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 15\, \lambda\sim CWmin\sim)\sqrt{2}\sqrt{5}}{20\sqrt{l\sim}} \right) \Bigg)$$

$$p\sim^{5} + \Bigg( \frac{1}{2} \operatorname{erf}\left( \frac{(8\, l\sim + 6\, \lambda\sim ACK_Timeout\sim + 8\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 7\, \lambda\sim CWmin\sim)\sqrt{2}}{8\sqrt{l\sim}} \right)$$

$$- \frac{1}{2} \operatorname{erf}\left( \frac{(10\, l\sim + 8\, \lambda\sim ACK_Timeout\sim + 10\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 15\, \lambda\sim CWmin\sim)\sqrt{2}\sqrt{5}}{20\sqrt{l\sim}} \right) \Bigg)$$

$$p\sim^{4} + \Bigg( - \frac{1}{2} \operatorname{erf}\left( \frac{(8\, l\sim + 6\, \lambda\sim ACK_Timeout\sim + 8\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 7\, \lambda\sim CWmin\sim)\sqrt{2}}{8\sqrt{l\sim}} \right)$$

$$+ \frac{1}{2} \operatorname{erf}\left( \frac{(6\, l\sim + 4\, \lambda\sim ACK_Timeout\sim + 6\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 3\, \lambda\sim CWmin\sim)\sqrt{2}\sqrt{3}}{12\sqrt{l\sim}} \right) \Bigg) p\sim^{3}$$

$$+ \Bigg( - \frac{1}{2} \operatorname{erf}\left( \frac{(6\, l\sim + 4\, \lambda\sim ACK_Timeout\sim + 6\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 3\, \lambda\sim CWmin\sim)\sqrt{2}\sqrt{3}}{12\sqrt{l\sim}} \right)$$

$$+ \frac{1}{2} \operatorname{erf}\left( \frac{4\, l\sim + 2\, \lambda\sim ACK_Timeout\sim + 4\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + \lambda\sim CWmin\sim}{4\sqrt{l\sim}} \right) \Bigg) p\sim^{2}$$

$$+ \Bigg( \frac{1}{2} \operatorname{erf}\left( \frac{(\lambda\sim Ed_ACK\sim - \lambda\sim t\sim + \lambda\sim SIFS\sim + l\sim + \lambda\sim DIFS\sim)\sqrt{2}}{2\sqrt{l\sim}} \right)$$

$$- \frac{1}{2} \operatorname{erf}\left( \frac{4\, l\sim + 2\, \lambda\sim ACK_Timeout\sim + 4\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + \lambda\sim CWmin\sim}{4 \sqrt{l\sim}} \right) \Bigg) p\sim + \frac{1}{2}$$

$$- \frac{1}{2} \operatorname{erf}\left( \frac{(\lambda\sim Ed_ACK\sim - \lambda\sim t\sim + \lambda\sim SIFS\sim + l\sim + \lambda\sim DIFS\sim) \sqrt{2}}{2 \sqrt{l\sim}} \right)$$

```
> #sa densité de probabilité :
  fTRd_k:=diff(FTRd_k,t);
```

$$fTRd_k := \Bigg( \frac{1}{4} \frac{e^{-\frac{(16\, l\sim + 14\, \lambda\sim ACK_Timeout\sim + 16\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 127\, \lambda\sim CWmin\sim)^2}{64\, l\sim}} \lambda\sim}{\sqrt{\pi} \sqrt{l\sim}}$$

$$- \frac{1}{14} \frac{e^{-\frac{(14\, l\sim + 12\, \lambda\sim ACK_Timeout\sim + 14\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 63\, \lambda\sim CWmin\sim)^2}{56\, l\sim}} \lambda\sim \sqrt{2} \sqrt{7}}{\sqrt{\pi} \sqrt{l\sim}} \Bigg) p\sim^7$$

$$+ \Bigg( \frac{1}{14} \frac{e^{-\frac{(14\, l\sim + 12\, \lambda\sim ACK_Timeout\sim + 14\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 63\, \lambda\sim CWmin\sim)^2}{56\, l\sim}} \lambda\sim \sqrt{2} \sqrt{7}}{\sqrt{\pi} \sqrt{l\sim}}$$

$$- \frac{1}{6} \frac{e^{-\frac{(12\, l\sim + 10\, \lambda\sim ACK_Timeout\sim + 12\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 31\, \lambda\sim CWmin\sim)^2}{48\, l\sim}} \lambda\sim \sqrt{3}}{\sqrt{\pi} \sqrt{l\sim}} \Bigg) p\sim^6$$

$$+ \Bigg( \frac{1}{6} \frac{e^{-\frac{(12\, l\sim + 10\, \lambda\sim ACK_Timeout\sim + 12\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 31\, \lambda\sim CWmin\sim)^2}{48\, l\sim}} \lambda\sim \sqrt{3}}{\sqrt{\pi} \sqrt{l\sim}}$$

$$- \frac{1}{10} \frac{e^{-\frac{(10\, l\sim + 8\, \lambda\sim ACK_Timeout\sim + 10\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + 15\, \lambda\sim CWmin\sim)^2}{40\, l\sim}} \lambda\sim \sqrt{2} \sqrt{5}}{\sqrt{\pi} \sqrt{l\sim}} \Bigg) p\sim^5 + \Bigg($$

(7

$$-\frac{1}{4}\,\frac{e^{-\frac{(8\,l\sim+6\,\lambda\sim ACK_Timeout\sim+8\,\lambda\sim DIFS\sim+2\,\lambda\sim Ed_ACK\sim-2\,\lambda\sim t\sim+2\,\lambda\sim SIFS\sim+7\,\lambda\sim CWmin\sim)^2}{32\,l\sim}}\,\lambda\sim\sqrt{2}}{\sqrt{\pi}\,\sqrt{l\sim}}$$

$$+\frac{1}{10}\,\frac{e^{-\frac{(10\,l\sim+8\,\lambda\sim ACK_Timeout\sim+10\,\lambda\sim DIFS\sim+2\,\lambda\sim Ed_ACK\sim-2\,\lambda\sim t\sim+2\,\lambda\sim SIFS\sim+15\,\lambda\sim CWmin\sim)^2}{40\,l\sim}}\,\lambda\sim\sqrt{2}\,\sqrt{5}}{\sqrt{\pi}\,\sqrt{l\sim}}\Bigg)\,p\sim^4$$

$$+\Bigg(\frac{1}{4}\,\frac{e^{-\frac{(8\,l\sim+6\,\lambda\sim ACK_Timeout\sim+8\,\lambda\sim DIFS\sim+2\,\lambda\sim Ed_ACK\sim-2\,\lambda\sim t\sim+2\,\lambda\sim SIFS\sim+7\,\lambda\sim CWmin\sim)^2}{32\,l\sim}}\,\lambda\sim\sqrt{2}}{\sqrt{\pi}\,\sqrt{l\sim}}$$

$$-\frac{1}{6}\,\frac{e^{-\frac{(6\,l\sim+4\,\lambda\sim ACK_Timeout\sim+6\,\lambda\sim DIFS\sim+2\,\lambda\sim Ed_ACK\sim-2\,\lambda\sim t\sim+2\,\lambda\sim SIFS\sim+3\,\lambda\sim CWmin\sim)^2}{24\,l\sim}}\,\lambda\sim\sqrt{2}\,\sqrt{3}}{\sqrt{\pi}\,\sqrt{l\sim}}\Bigg)\,p\sim^3$$

$$+\Bigg(\frac{1}{6}\,\frac{e^{-\frac{(6\,l\sim+4\,\lambda\sim ACK_Timeout\sim+6\,\lambda\sim DIFS\sim+2\,\lambda\sim Ed_ACK\sim-2\,\lambda\sim t\sim+2\,\lambda\sim SIFS\sim+3\,\lambda\sim CWmin\sim)^2}{24\,l\sim}}\,\lambda\sim\sqrt{2}\,\sqrt{3}}{\sqrt{\pi}\,\sqrt{l\sim}}$$

$$-\frac{1}{2}\,\frac{e^{-\frac{(4\,l\sim+2\,\lambda\sim ACK_Timeout\sim+4\,\lambda\sim DIFS\sim+2\,\lambda\sim Ed_ACK\sim-2\,\lambda\sim t\sim+2\,\lambda\sim SIFS\sim+\lambda\sim CWmin\sim)^2}{16\,l\sim}}\,\lambda\sim}{\sqrt{\pi}\,\sqrt{l\sim}}\Bigg)\,p\sim^2+\Bigg($$

$$-\frac{1}{2}\,\frac{e^{-\frac{(\lambda\sim Ed_ACK\sim-\lambda\sim t\sim+\lambda\sim SIFS\sim+l\sim+\lambda\sim DIFS\sim)^2}{2\,l\sim}}\,\lambda\sim\sqrt{2}}{\sqrt{\pi}\,\sqrt{l\sim}}$$

$$+ \frac{1}{2} \frac{e^{-\frac{(4\, l\sim + 2\, \lambda\sim ACK_Timeout\sim + 4\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim - 2\, \lambda\sim t\sim + 2\, \lambda\sim SIFS\sim + \lambda\sim CWmin\sim)^2}{16\, l\sim}} \lambda\sim}{\sqrt{\pi}\, \sqrt{l\sim}} \Bigg) p\sim$$

$$+ \frac{1}{2} \frac{e^{-\frac{(\lambda\sim Ed_ACK\sim - \lambda\sim t\sim + \lambda\sim SIFS\sim + l\sim + \lambda\sim DIFS\sim)^2}{2\, l\sim}} \lambda\sim \sqrt{2}}{\sqrt{\pi}\, \sqrt{l\sim}}$$

```
> #sa moyenne :
  muTRd_k:=unapply(simplify(int(t*fTRd_k,t=-infinity..infinity)),l);
```

$$muTRd_k := l\sim \rightarrow \frac{1}{2} \frac{1}{\lambda\sim} \big(2\, \lambda\sim Ed_ACK\sim + 2\, \lambda\sim DIFS\sim + 2\, \lambda\sim SIFS\sim + 2\, p\sim^6\, DIFS\sim \lambda\sim + 2\, p\sim^5\, ACK_Timeout\sim \lambda\sim + 2\, p\sim^3\, DIFS\sim \lambda\sim \tag{8}$$

$$+ 2\, p\sim^3\, ACK_Timeout\sim \lambda\sim + 4\, p\sim^3\, CWmin\sim \lambda\sim + 2\, p\sim^7\, DIFS\sim \lambda\sim + 64\, p\sim^7\, CWmin\sim \lambda\sim + 2\, p\sim^7\, ACK_Timeout\sim \lambda\sim$$

$$+ 2\, p\sim^4\, ACK_Timeout\sim \lambda\sim + 8\, p\sim^4\, CWmin\sim \lambda\sim + 2\, p\sim^4\, DIFS\sim \lambda\sim + 2\, p\sim ACK_Timeout\sim \lambda\sim + p\sim CWmin\sim \lambda\sim + 2\, p\sim DIFS\sim \lambda\sim$$

$$+ 2\, p\sim^7\, l\sim + 2\, p\sim^2\, ACK_Timeout\sim \lambda\sim + 2\, p\sim^2\, CWmin\sim \lambda\sim + 2\, p\sim^2\, DIFS\sim \lambda\sim + 16\, p\sim^5\, CWmin\sim \lambda\sim + 2\, p\sim^5\, DIFS\sim \lambda\sim$$

$$+ 2\, p\sim^6\, ACK_Timeout\sim \lambda\sim + 32\, p\sim^6\, CWmin\sim \lambda\sim + 2\, p\sim^5\, l\sim + 2\, p\sim^3\, l\sim + 2\, p\sim l\sim + 2\, p\sim^4\, l\sim + 2\, p\sim^2\, l\sim + 2\, l\sim + 2\, p\sim^6\, l\sim\big)$$

```
> #sa variance :
  vTRd_k:=unapply(simplify(int(t^2*fTRd_k,t=-infinity..infinity))-muTRd_k(l)^2,l);
```

$$vTRd_k := l\sim \rightarrow \frac{1}{4} \frac{1}{\lambda\sim^2} \big(4\, p\sim^7\, l\sim + 4\, p\sim^5\, l\sim + 4\, p\sim^3\, l\sim + 4\, p\sim l\sim + 4\, p\sim^4\, l\sim + 4\, p\sim^2\, l\sim + 4\, l\sim^2 + 36\, p\sim^4\, l\sim^2 + 12\, p\sim l\sim^2 + 20\, p\sim^2\, l\sim^2 \tag{9}$$

$$+ 8\, l\sim \lambda\sim DIFS\sim + 8\, l\sim \lambda\sim Ed_ACK\sim + 8\, l\sim \lambda\sim SIFS\sim + 16\, p\sim l\sim ACK_Timeout\sim \lambda\sim + 40\, p\sim^2\, l\sim DIFS\sim \lambda\sim$$

$$+ 32\, p\sim^2\, l\sim ACK_Timeout\sim \lambda\sim + 28\, p\sim^2\, l\sim CWmin\sim \lambda\sim + 8\, p\sim^2\, l\sim SIFS\sim \lambda\sim + 8\, p\sim^2\, l\sim Ed_ACK\sim \lambda\sim + 2300\, p\sim^7\, l\sim CWmin\sim \lambda\sim$$

$$+ 120\, p\sim^7\, l\sim DIFS\sim \lambda\sim + 8\, p\sim l\sim CWmin\sim \lambda\sim + 80\, p\sim^5\, l\sim ACK_Timeout\sim \lambda\sim + 28\, p\sim^3\, l\sim^2 + 60\, p\sim^7\, l\sim^2 + 44\, p\sim^5\, l\sim^2$$

$$+ 8\, p\sim^5\, l\sim Ed_ACK\sim \lambda\sim + 8\, p\sim^6\, l\sim Ed_ACK\sim \lambda\sim + 8\, p\sim^7\, l\sim Ed_ACK\sim \lambda\sim + 48\, p\sim^3\, l\sim ACK_Timeout\sim \lambda\sim + 8\, p\sim^5\, l\sim SIFS\sim \lambda\sim$$

$$+ 56\, p\sim^3\, l\sim DIFS\sim \lambda\sim + 444\, p\sim^5\, l\sim CWmin\sim \lambda\sim + 88\, p\sim^5\, l\sim DIFS\sim \lambda\sim + 76\, p\sim^3\, l\sim CWmin\sim \lambda\sim + 8\, p\sim^3\, l\sim SIFS\sim \lambda\sim$$

$$+ 8\, p\sim^{3}\, l\sim Ed_ACK\sim \lambda\sim + 52\, p\sim^{6}\, l\sim^{2} + 8\, p\sim^{4}\, l\sim SIFS\sim \lambda\sim + 8\, p\sim^{4}\, l\sim Ed_ACK\sim \lambda\sim + 96\, p\sim^{6}\, l\sim ACK_Timeout\sim \lambda\sim$$

$$+ 8\, p\sim^{6}\, l\sim SIFS\sim \lambda\sim + 1020\, p\sim^{6}\, l\sim CWmin\sim \lambda\sim + 104\, p\sim^{6}\, l\sim DIFS\sim \lambda\sim + 188\, p\sim^{4}\, l\sim CWmin\sim \lambda\sim + 72\, p\sim^{4}\, l\sim DIFS\sim \lambda\sim$$

$$+ 64\, p\sim^{4}\, l\sim ACK_Timeout\sim \lambda\sim + 8\, p\sim l\sim SIFS\sim \lambda\sim + 8\, p\sim l\sim Ed_ACK\sim \lambda\sim + 24\, p\sim l\sim DIFS\sim \lambda\sim + 8\, p\sim^{7}\, l\sim SIFS\sim \lambda\sim$$

$$+ 112\, p\sim^{7}\, l\sim ACK_Timeout\sim \lambda\sim + 8\, p\sim^{3}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{3}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim^{3}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2}$$

$$+ 32\, p\sim^{4}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim SIFS\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 4\, p\sim SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 8\, p\sim ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 4\, p\sim ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim CWmin\sim DIFS\sim \lambda\sim^{2}$$

$$+ 64\, p\sim^{5}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 4\, p\sim Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{2}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{2}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 28\, p\sim^{2}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2}$$

$$+ 32\, p\sim^{2}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 20\, p\sim^{2}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{2}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 2300\, p\sim^{7}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{7}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 256\, p\sim^{7}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{7}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 256\, p\sim^{7}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 64\, p\sim^{4}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2}$$

$$+ 112\, p\sim^{7}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 2044\, p\sim^{7}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 156\, p\sim^{4}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 188\, p\sim^{4}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 32\, p\sim^{4}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{4}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 8\, p\sim Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{5}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 444\, p\sim^{5}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 64\, p\sim^{5}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{6}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 128\, p\sim^{6}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2}$$

$$+ 892\, p\sim^{6}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 380\, p\sim^{5}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 96\, p\sim^{6}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{6}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 128\, p\sim^{6}\, SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 1020\, p\sim^6\, CWmin\sim DIFS\sim \lambda\sim^2 + 48\, p\sim^3\, ACK_Timeout\sim DIFS\sim \lambda\sim^2 + 16\, p\sim^3\, SIFS\sim CWmin\sim \lambda\sim^2$$

$$+ 8\, p\sim^3\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^2 + 60\, p\sim^3\, ACK_Timeout\sim CWmin\sim \lambda\sim^2 + 76\, p\sim^3\, CWmin\sim DIFS\sim \lambda\sim^2$$

$$+ 8\, p\sim^3\, ACK_Timeout\sim SIFS\sim \lambda\sim^2 + 4\, l\sim + 80\, p\sim^5\, ACK_Timeout\sim DIFS\sim \lambda\sim^2 + 20\, p\sim^2\, DIFS\sim^2\, \lambda\sim^2 + 12\, p\sim DIFS\sim^2\, \lambda\sim^2$$

$$+ 36\, p\sim^4\, DIFS\sim^2\, \lambda\sim^2 + 8\, p\sim^2\, CWmin\sim^2\, \lambda\sim^2 + 60\, p\sim^7\, DIFS\sim^2\, \lambda\sim^2 + 12160\, p\sim^7\, CWmin\sim^2\, \lambda\sim^2 + 52\, p\sim^7\, ACK_Timeout\sim^2\, \lambda\sim^2$$

$$+ 20\, p\sim^3\, ACK_Timeout\sim^2\, \lambda\sim^2 + 40\, p\sim^3\, CWmin\sim^2\, \lambda\sim^2 + 28\, p\sim^3\, DIFS\sim^2\, \lambda\sim^2 + 736\, p\sim^5\, CWmin\sim^2\, \lambda\sim^2 + 44\, p\sim^5\, DIFS\sim^2\, \lambda\sim^2$$

$$+ 36\, p\sim^5\, ACK_Timeout\sim^2\, \lambda\sim^2 + 3008\, p\sim^6\, CWmin\sim^2\, \lambda\sim^2 + 52\, p\sim^6\, DIFS\sim^2\, \lambda\sim^2 + 44\, p\sim^6\, ACK_Timeout\sim^2\, \lambda\sim^2$$

$$+ 176\, p\sim^4\, CWmin\sim^2\, \lambda\sim^2 + 28\, p\sim^4\, ACK_Timeout\sim^2\, \lambda\sim^2 + p\sim CWmin\sim^2\, \lambda\sim^2 + 4\, p\sim ACK_Timeout\sim^2\, \lambda\sim^2$$

$$+ 12\, p\sim^2\, ACK_Timeout\sim^2\, \lambda\sim^2 + 8\, \lambda\sim^2\, DIFS\sim SIFS\sim + 8\, \lambda\sim^2\, Ed_ACK\sim SIFS\sim + 4\, \lambda\sim^2\, DIFS\sim^2 + 4\, \lambda\sim^2\, Ed_ACK\sim^2 + 4\, \lambda\sim^2\, SIFS\sim^2$$

$$+ 8\, \lambda\sim^2\, DIFS\sim Ed_ACK\sim + 4\, p\sim^6\, l\sim\Big) - \frac{1}{4}\, \frac{1}{\lambda\sim^2}\Big(2\, \lambda\sim Ed_ACK\sim + 2\, \lambda\sim DIFS\sim + 2\, \lambda\sim SIFS\sim + 2\, p\sim^6\, DIFS\sim \lambda\sim$$

$$+ 2\, p\sim^5\, ACK_Timeout\sim \lambda\sim + 2\, p\sim^3\, DIFS\sim \lambda\sim + 2\, p\sim^3\, ACK_Timeout\sim \lambda\sim + 4\, p\sim^3\, CWmin\sim \lambda\sim + 2\, p\sim^7\, DIFS\sim \lambda\sim$$

$$+ 64\, p\sim^7\, CWmin\sim \lambda\sim + 2\, p\sim^7\, ACK_Timeout\sim \lambda\sim + 2\, p\sim^4\, ACK_Timeout\sim \lambda\sim + 8\, p\sim^4\, CWmin\sim \lambda\sim + 2\, p\sim^4\, DIFS\sim \lambda\sim$$

$$+ 2\, p\sim ACK_Timeout\sim \lambda\sim + p\sim CWmin\sim \lambda\sim + 2\, p\sim DIFS\sim \lambda\sim + 2\, p\sim^7\, l\sim + 2\, p\sim^2\, ACK_Timeout\sim \lambda\sim + 2\, p\sim^2\, CWmin\sim \lambda\sim$$

$$+ 2\, p\sim^2\, DIFS\sim \lambda\sim + 16\, p\sim^5\, CWmin\sim \lambda\sim + 2\, p\sim^5\, DIFS\sim \lambda\sim + 2\, p\sim^6\, ACK_Timeout\sim \lambda\sim + 32\, p\sim^6\, CWmin\sim \lambda\sim + 2\, p\sim^5\, l\sim + 2\, p\sim^3\, l\sim$$

$$+ 2\, p\sim l\sim + 2\, p\sim^4\, l\sim + 2\, p\sim^2\, l\sim + 2\, l\sim + 2\, p\sim^6\, l\sim\Big)^2$$

```
> #sur un lien à sauts multiples de longueur hops :
  #temps d'un aller-retour, son écart type :
  sROTd:=sqrt(hops*(vTRd_k(l1)+vTRd_k(l2)));
```

$sROTd :=$ **(10**

$$\Bigg(hops\sim\Bigg(\frac{1}{4\, \lambda\sim^2}\Big(1020\, p\sim^6\, l1\sim CWmin\sim \lambda\sim + 104\, p\sim^6\, l1\sim DIFS\sim \lambda\sim + 188\, p\sim^4\, l1\sim CWmin\sim \lambda\sim + 72\, p\sim^4\, l1\sim DIFS\sim \lambda\sim$$

$$+ 64\, p\sim^{4}\, l1\sim ACK_Timeout\sim \lambda\sim + 8\, p\sim l1\sim SIFS\sim \lambda\sim + 8\, p\sim l1\sim Ed_ACK\sim \lambda\sim + 24\, p\sim l1\sim DIFS\sim \lambda\sim + 8\, p\sim^{7}\, l1\sim SIFS\sim \lambda\sim$$

$$+ 112\, p\sim^{7}\, l1\sim ACK_Timeout\sim \lambda\sim + 88\, p\sim^{5}\, l1\sim DIFS\sim \lambda\sim + 76\, p\sim^{3}\, l1\sim CWmin\sim \lambda\sim + 8\, p\sim^{3}\, l1\sim SIFS\sim \lambda\sim + 8\, p\sim^{3}\, l1\sim Ed_ACK\sim \lambda\sim$$

$$+ 8\, p\sim^{4}\, l1\sim SIFS\sim \lambda\sim + 8\, p\sim^{4}\, l1\sim Ed_ACK\sim \lambda\sim + 96\, p\sim^{6}\, l1\sim ACK_Timeout\sim \lambda\sim + 8\, p\sim^{6}\, l1\sim SIFS\sim \lambda\sim + 8\, p\sim^{5}\, l1\sim Ed_ACK\sim \lambda\sim$$

$$+ 8\, p\sim^{6}\, l1\sim Ed_ACK\sim \lambda\sim + 8\, p\sim^{7}\, l1\sim Ed_ACK\sim \lambda\sim + 48\, p\sim^{3}\, l1\sim ACK_Timeout\sim \lambda\sim + 8\, p\sim^{5}\, l1\sim SIFS\sim \lambda\sim + 56\, p\sim^{3}\, l1\sim DIFS\sim \lambda\sim$$

$$+ 444\, p\sim^{5}\, l1\sim CWmin\sim \lambda\sim + 4\, l1\sim^{2} + 28\, p\sim^{2}\, l1\sim CWmin\sim \lambda\sim + 8\, p\sim^{2}\, l1\sim SIFS\sim \lambda\sim + 8\, p\sim^{2}\, l1\sim Ed_ACK\sim \lambda\sim$$

$$+ 2300\, p\sim^{7}\, l1\sim CWmin\sim \lambda\sim + 120\, p\sim^{7}\, l1\sim DIFS\sim \lambda\sim + 8\, p\sim l1\sim CWmin\sim \lambda\sim + 80\, p\sim^{5}\, l1\sim ACK_Timeout\sim \lambda\sim + 8\, l1\sim \lambda\sim DIFS\sim$$

$$+ 8\, l1\sim \lambda\sim Ed_ACK\sim + 8\, l1\sim \lambda\sim SIFS\sim + 16\, p\sim l1\sim ACK_Timeout\sim \lambda\sim + 40\, p\sim^{2}\, l1\sim DIFS\sim \lambda\sim + 32\, p\sim^{2}\, l1\sim ACK_Timeout\sim \lambda\sim$$

$$+ 8\, p\sim^{3}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{3}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim^{3}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 32\, p\sim^{4}\, SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 8\, p\sim SIFS\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 4\, p\sim SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 4\, p\sim ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim CWmin\sim DIFS\sim \lambda\sim^{2} + 64\, p\sim^{5}\, SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 4\, p\sim Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{2}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 28\, p\sim^{2}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2}$$

$$+ 32\, p\sim^{2}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 20\, p\sim^{2}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{2}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 2300\, p\sim^{7}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{7}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 256\, p\sim^{7}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{7}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 256\, p\sim^{7}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 64\, p\sim^{4}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2}$$

$$+ 112\, p\sim^{7}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 2044\, p\sim^{7}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 156\, p\sim^{4}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 188\, p\sim^{4}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 32\, p\sim^{4}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{4}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 8\, p\sim Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{5}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 444\, p\sim^{5}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 64\, p\sim^{5}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{6}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 128\, p\sim^{6}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2}$$

$$+ 892\, p\sim^{6}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 380\, p\sim^{5}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 96\, p\sim^{6}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 8\, p\sim^{6}\, ACK_Timeout\sim\, SIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{6}\, ACK_Timeout\sim\, Ed_ACK\sim\, \lambda\sim^{2} + 128\, p\sim^{6}\, SIFS\sim\, CWmin\sim\, \lambda\sim^{2}$$

$$+ 1020\, p\sim^{6}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2} + 48\, p\sim^{3}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2} + 16\, p\sim^{3}\, SIFS\sim\, CWmin\sim\, \lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, ACK_Timeout\sim\, Ed_ACK\sim\, \lambda\sim^{2} + 60\, p\sim^{3}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 76\, p\sim^{3}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, ACK_Timeout\sim\, SIFS\sim\, \lambda\sim^{2} + 80\, p\sim^{5}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2} + 4\, p\sim\, l1\sim + 4\, p\sim^{4}\, l1\sim + 4\, p\sim^{2}\, l1\sim + 36\, p\sim^{4}\, l1\sim^{2}$$

$$+ 12\, p\sim\, l1\sim^{2} + 20\, p\sim^{2}\, l1\sim^{2} + 28\, p\sim^{3}\, l1\sim^{2} + 60\, p\sim^{7}\, l1\sim^{2} + 44\, p\sim^{5}\, l1\sim^{2} + 52\, p\sim^{6}\, l1\sim^{2} + 4\, p\sim^{6}\, l1\sim + 4\, p\sim^{5}\, l1\sim + 4\, p\sim^{3}\, l1\sim$$

$$+ 20\, p\sim^{2}\, DIFS\sim^{2}\, \lambda\sim^{2} + 4\, p\sim^{7}\, l1\sim + 12\, p\sim\, DIFS\sim^{2}\, \lambda\sim^{2} + 36\, p\sim^{4}\, DIFS\sim^{2}\, \lambda\sim^{2} + 8\, p\sim^{2}\, CWmin\sim^{2}\, \lambda\sim^{2} + 60\, p\sim^{7}\, DIFS\sim^{2}\, \lambda\sim^{2}$$

$$+ 12160\, p\sim^{7}\, CWmin\sim^{2}\, \lambda\sim^{2} + 52\, p\sim^{7}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 20\, p\sim^{3}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 40\, p\sim^{3}\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 28\, p\sim^{3}\, DIFS\sim^{2}\, \lambda\sim^{2} + 736\, p\sim^{5}\, CWmin\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{5}\, DIFS\sim^{2}\, \lambda\sim^{2} + 36\, p\sim^{5}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 3008\, p\sim^{6}\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 52\, p\sim^{6}\, DIFS\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{6}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 176\, p\sim^{4}\, CWmin\sim^{2}\, \lambda\sim^{2} + 28\, p\sim^{4}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + p\sim\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 4\, p\sim ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 12\, p\sim^{2}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 8\, \lambda\sim^{2}\, DIFS\sim SIFS\sim + 8\, \lambda\sim^{2}\, Ed_ACK\sim SIFS\sim + 4\, \lambda\sim^{2}\, DIFS\sim^{2}$$

$$+ 4\, \lambda\sim^{2}\, Ed_ACK\sim^{2} + 4\, \lambda\sim^{2}\, SIFS\sim^{2} + 8\, \lambda\sim^{2}\, DIFS\sim Ed_ACK\sim + 4\, l1\sim\Big) - \frac{1}{4\, \lambda\sim^{2}}\Big(2\, \lambda\sim Ed_ACK\sim + 2\, \lambda\sim DIFS\sim + 2\, \lambda\sim SIFS\sim$$

$$+ 2\, p\sim^{6}\, DIFS\sim\lambda\sim + 2\, p\sim^{5}\, ACK_Timeout\sim\lambda\sim + 2\, p\sim^{3}\, DIFS\sim\lambda\sim + 2\, p\sim^{3}\, ACK_Timeout\sim\lambda\sim + 4\, p\sim^{3}\, CWmin\sim\lambda\sim + 2\, p\sim^{7}\, DIFS\sim\lambda\sim$$

$$+ 64\, p\sim^{7}\, CWmin\sim\lambda\sim + 2\, p\sim^{7}\, ACK_Timeout\sim\lambda\sim + 2\, p\sim^{4}\, ACK_Timeout\sim\lambda\sim + 8\, p\sim^{4}\, CWmin\sim\lambda\sim + 2\, p\sim^{4}\, DIFS\sim\lambda\sim$$

$$+ 2\, p\sim ACK_Timeout\sim\lambda\sim + p\sim CWmin\sim\lambda\sim + 2\, p\sim DIFS\sim\lambda\sim + 2\, p\sim^{2}\, ACK_Timeout\sim\lambda\sim + 2\, p\sim^{2}\, CWmin\sim\lambda\sim + 2\, p\sim^{2}\, DIFS\sim\lambda\sim$$

$$+ 16\, p\sim^{5}\, CWmin\sim\lambda\sim + 2\, p\sim^{5}\, DIFS\sim\lambda\sim + 2\, p\sim^{6}\, ACK_Timeout\sim\lambda\sim + 32\, p\sim^{6}\, CWmin\sim\lambda\sim + 2\, p\sim l1\sim + 2\, p\sim^{4}\, l1\sim + 2\, p\sim^{2}\, l1\sim$$

$$+ 2\, p\sim^{6}\, l1\sim + 2\, p\sim^{5}\, l1\sim + 2\, p\sim^{3}\, l1\sim + 2\, p\sim^{7}\, l1\sim + 2\, l1\sim\Big)^{2} + \frac{1}{4\, \lambda\sim^{2}}\Big(8\, p\sim l2\sim Ed_ACK\sim\lambda\sim + 24\, p\sim l2\sim DIFS\sim\lambda\sim$$

$$+ 8\, p\sim^{7}\, l2\sim SIFS\sim\lambda\sim + 112\, p\sim^{7}\, l2\sim ACK_Timeout\sim\lambda\sim + 104\, p\sim^{6}\, l2\sim DIFS\sim\lambda\sim + 188\, p\sim^{4}\, l2\sim CWmin\sim\lambda\sim + 72\, p\sim^{4}\, l2\sim DIFS\sim\lambda\sim$$

$$+ 64\, p\sim^{4}\, l2\sim ACK_Timeout\sim\lambda\sim + 8\, p\sim l2\sim SIFS\sim\lambda\sim + 76\, p\sim^{3}\, l2\sim CWmin\sim\lambda\sim + 8\, p\sim^{3}\, l2\sim SIFS\sim\lambda\sim + 8\, p\sim^{3}\, l2\sim Ed_ACK\sim\lambda\sim$$

$$+ 8\, p\sim^{4}\, l2\sim SIFS\sim\lambda\sim + 8\, p\sim^{4}\, l2\sim Ed_ACK\sim\lambda\sim + 96\, p\sim^{6}\, l2\sim ACK_Timeout\sim\lambda\sim + 8\, p\sim^{6}\, l2\sim SIFS\sim\lambda\sim + 1020\, p\sim^{6}\, l2\sim CWmin\sim\lambda\sim$$

$$+ 8\, p\sim^{5}\, l2\sim SIFS\sim \lambda\sim + 56\, p\sim^{3}\, l2\sim DIFS\sim \lambda\sim + 444\, p\sim^{5}\, l2\sim CWmin\sim \lambda\sim + 88\, p\sim^{5}\, l2\sim DIFS\sim \lambda\sim + 16\, p\sim l2\sim ACK_Timeout\sim \lambda\sim$$

$$+ 40\, p\sim^{2}\, l2\sim DIFS\sim \lambda\sim + 32\, p\sim^{2}\, l2\sim ACK_Timeout\sim \lambda\sim + 80\, p\sim^{5}\, l2\sim ACK_Timeout\sim \lambda\sim + 8\, p\sim^{5}\, l2\sim Ed_ACK\sim \lambda\sim$$

$$+ 8\, p\sim^{6}\, l2\sim Ed_ACK\sim \lambda\sim + 8\, p\sim^{7}\, l2\sim Ed_ACK\sim \lambda\sim + 48\, p\sim^{3}\, l2\sim ACK_Timeout\sim \lambda\sim + 2300\, p\sim^{7}\, l2\sim CWmin\sim \lambda\sim$$

$$+ 120\, p\sim^{7}\, l2\sim DIFS\sim \lambda\sim + 8\, p\sim l2\sim CWmin\sim \lambda\sim + 28\, p\sim^{3}\, l2\sim^{2} + 28\, p\sim^{2}\, l2\sim CWmin\sim \lambda\sim + 8\, p\sim^{2}\, l2\sim SIFS\sim \lambda\sim$$

$$+ 8\, p\sim^{2}\, l2\sim Ed_ACK\sim \lambda\sim + 4\, l2\sim^{2} + 8\, l2\sim \lambda\sim DIFS\sim + 8\, l2\sim \lambda\sim Ed_ACK\sim + 8\, l2\sim \lambda\sim SIFS\sim + 8\, p\sim^{3}\, SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim^{3}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 32\, p\sim^{4}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 16\, p\sim ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 4\, p\sim SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 4\, p\sim ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim CWmin\sim DIFS\sim \lambda\sim^{2} + 64\, p\sim^{5}\, SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 4\, p\sim Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{2}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 28\, p\sim^{2}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2}$$

$$+ 32\, p\sim^{2}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 20\, p\sim^{2}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{2}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 2300\, p\sim^{7}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{7}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 256\, p\sim^{7}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{7}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 256\, p\sim^{7}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 64\, p\sim^{4}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2}$$

$$+ 112\, p\sim^{7}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 2044\, p\sim^{7}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 156\, p\sim^{4}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 188\, p\sim^{4}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 32\, p\sim^{4}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{4}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 8\, p\sim Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{5}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 444\, p\sim^{5}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 64\, p\sim^{5}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{6}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 128\, p\sim^{6}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2}$$

$$+ 892\, p\sim^{6}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 380\, p\sim^{5}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 96\, p\sim^{6}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{6}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 128\, p\sim^{6}\, SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 1020\, p\sim^{6}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2} + 48\, p\sim^{3}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2} + 16\, p\sim^{3}\, SIFS\sim\, CWmin\sim\, \lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, ACK_Timeout\sim\, Ed_ACK\sim\, \lambda\sim^{2} + 60\, p\sim^{3}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 76\, p\sim^{3}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, ACK_Timeout\sim\, SIFS\sim\, \lambda\sim^{2} + 80\, p\sim^{5}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2} + 52\, p\sim^{6}\, l2\sim^{2} + 4\, p\sim^{6}\, l2\sim + 4\, p\sim^{2}\, l2\sim + 36\, p\sim^{4}\, l2\sim^{2}$$

$$+ 12\, p\sim\, l2\sim^{2} + 20\, p\sim^{2}\, l2\sim^{2} + 60\, p\sim^{7}\, l2\sim^{2} + 44\, p\sim^{5}\, l2\sim^{2} + 4\, p\sim^{7}\, l2\sim + 4\, p\sim^{5}\, l2\sim + 4\, p\sim^{3}\, l2\sim + 4\, p\sim\, l2\sim + 4\, p\sim^{4}\, l2\sim$$

$$+ 20\, p\sim^{2}\, DIFS\sim^{2}\, \lambda\sim^{2} + 12\, p\sim\, DIFS\sim^{2}\, \lambda\sim^{2} + 36\, p\sim^{4}\, DIFS\sim^{2}\, \lambda\sim^{2} + 8\, p\sim^{2}\, CWmin\sim^{2}\, \lambda\sim^{2} + 60\, p\sim^{7}\, DIFS\sim^{2}\, \lambda\sim^{2}$$

$$+ 12160\, p\sim^{7}\, CWmin\sim^{2}\, \lambda\sim^{2} + 52\, p\sim^{7}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 20\, p\sim^{3}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 40\, p\sim^{3}\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 28\, p\sim^{3}\, DIFS\sim^{2}\, \lambda\sim^{2} + 736\, p\sim^{5}\, CWmin\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{5}\, DIFS\sim^{2}\, \lambda\sim^{2} + 36\, p\sim^{5}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 3008\, p\sim^{6}\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 52\, p\sim^{6}\, DIFS\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{6}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 176\, p\sim^{4}\, CWmin\sim^{2}\, \lambda\sim^{2} + 28\, p\sim^{4}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + p\sim\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 4\, p\sim\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 12\, p\sim^{2}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 8\, \lambda\sim^{2}\, DIFS\sim\, SIFS\sim + 8\, \lambda\sim^{2}\, Ed_ACK\sim\, SIFS\sim + 4\, \lambda\sim^{2}\, DIFS\sim^{2}$$

$$+ 4\, \lambda\sim^{2}\, Ed_ACK\sim^{2} + 4\, \lambda\sim^{2}\, SIFS\sim^{2} + 8\, \lambda\sim^{2}\, DIFS\sim\, Ed_ACK\sim + 4\, l2\sim\big) - \frac{1}{4\, \lambda\sim^{2}}\big(2\, \lambda\sim\, Ed_ACK\sim + 2\, \lambda\sim\, DIFS\sim + 2\, \lambda\sim\, SIFS\sim$$

$$+ 2\,p\sim^{6}\,DIFS\sim\lambda\sim + 2\,p\sim^{5}\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^{3}\,DIFS\sim\lambda\sim + 2\,p\sim^{3}\,ACK_Timeout\sim\lambda\sim + 4\,p\sim^{3}\,CWmin\sim\lambda\sim + 2\,p\sim^{7}\,DIFS\sim\lambda\sim$$

$$+ 64\,p\sim^{7}\,CWmin\sim\lambda\sim + 2\,p\sim^{7}\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^{4}\,ACK_Timeout\sim\lambda\sim + 8\,p\sim^{4}\,CWmin\sim\lambda\sim + 2\,p\sim^{4}\,DIFS\sim\lambda\sim$$

$$+ 2\,p\sim\,ACK_Timeout\sim\lambda\sim + p\sim\,CWmin\sim\lambda\sim + 2\,p\sim\,DIFS\sim\lambda\sim + 2\,p\sim^{2}\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^{2}\,CWmin\sim\lambda\sim + 2\,p\sim^{2}\,DIFS\sim\lambda\sim$$

$$+ 16\,p\sim^{5}\,CWmin\sim\lambda\sim + 2\,p\sim^{5}\,DIFS\sim\lambda\sim + 2\,p\sim^{6}\,ACK_Timeout\sim\lambda\sim + 32\,p\sim^{6}\,CWmin\sim\lambda\sim + 2\,p\sim^{6}\,l2\sim + 2\,p\sim^{2}\,l2\sim + 2\,p\sim^{7}\,l2\sim$$

$$+ 2\,p\sim^{5}\,l2\sim + 2\,p\sim^{3}\,l2\sim + 2\,p\sim\,l2\sim + 2\,p\sim^{4}\,l2\sim + 2\,l2\sim)^{2}))^{1/2}$$

```
> #sa moyenne :
  muROTd:=hops*(muTRd_k(l1)+muTRd_k(l2));
```

$$muROTd := hops\sim\left(\frac{1}{2\,\lambda\sim}(2\,\lambda\sim\,Ed_ACK\sim + 2\,\lambda\sim\,DIFS\sim + 2\,\lambda\sim\,SIFS\sim + 2\,p\sim^{6}\,DIFS\sim\lambda\sim + 2\,p\sim^{5}\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^{3}\,DIFS\sim\lambda\sim \quad \textbf{(11}$$

$$+ 2\,p\sim^{3}\,ACK_Timeout\sim\lambda\sim + 4\,p\sim^{3}\,CWmin\sim\lambda\sim + 2\,p\sim^{7}\,DIFS\sim\lambda\sim + 64\,p\sim^{7}\,CWmin\sim\lambda\sim + 2\,p\sim^{7}\,ACK_Timeout\sim\lambda\sim$$

$$+ 2\,p\sim^{4}\,ACK_Timeout\sim\lambda\sim + 8\,p\sim^{4}\,CWmin\sim\lambda\sim + 2\,p\sim^{4}\,DIFS\sim\lambda\sim + 2\,p\sim\,ACK_Timeout\sim\lambda\sim + p\sim\,CWmin\sim\lambda\sim + 2\,p\sim\,DIFS\sim\lambda\sim$$

$$+ 2\,p\sim^{2}\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^{2}\,CWmin\sim\lambda\sim + 2\,p\sim^{2}\,DIFS\sim\lambda\sim + 16\,p\sim^{5}\,CWmin\sim\lambda\sim + 2\,p\sim^{5}\,DIFS\sim\lambda\sim$$

$$+ 2\,p\sim^{6}\,ACK_Timeout\sim\lambda\sim + 32\,p\sim^{6}\,CWmin\sim\lambda\sim + 2\,p\sim\,l1\sim + 2\,p\sim^{4}\,l1\sim + 2\,p\sim^{2}\,l1\sim + 2\,p\sim^{6}\,l1\sim + 2\,p\sim^{5}\,l1\sim + 2\,p\sim^{3}\,l1\sim$$

$$+ 2\,p\sim^{7}\,l1\sim + 2\,l1\sim) + \frac{1}{2\,\lambda\sim}(2\,\lambda\sim\,Ed_ACK\sim + 2\,\lambda\sim\,DIFS\sim + 2\,\lambda\sim\,SIFS\sim + 2\,p\sim^{6}\,DIFS\sim\lambda\sim + 2\,p\sim^{5}\,ACK_Timeout\sim\lambda\sim$$

$$+ 2\,p\sim^{3}\,DIFS\sim\lambda\sim + 2\,p\sim^{3}\,ACK_Timeout\sim\lambda\sim + 4\,p\sim^{3}\,CWmin\sim\lambda\sim + 2\,p\sim^{7}\,DIFS\sim\lambda\sim + 64\,p\sim^{7}\,CWmin\sim\lambda\sim$$

$$+ 2\,p\sim^{7}\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^{4}\,ACK_Timeout\sim\lambda\sim + 8\,p\sim^{4}\,CWmin\sim\lambda\sim + 2\,p\sim^{4}\,DIFS\sim\lambda\sim + 2\,p\sim\,ACK_Timeout\sim\lambda\sim$$

$$+ p\sim CWmin\sim\lambda\sim + 2\,p\sim DIFS\sim\lambda\sim + 2\,p\sim^{2} ACK_Timeout\sim\lambda\sim + 2\,p\sim^{2} CWmin\sim\lambda\sim + 2\,p\sim^{2} DIFS\sim\lambda\sim + 16\,p\sim^{5} CWmin\sim\lambda\sim$$

$$+ 2\,p\sim^{5} DIFS\sim\lambda\sim + 2\,p\sim^{6} ACK_Timeout\sim\lambda\sim + 32\,p\sim^{6} CWmin\sim\lambda\sim + 2\,p\sim^{6} l2\sim + 2\,p\sim^{2} l2\sim + 2\,p\sim^{7} l2\sim + 2\,p\sim^{5} l2\sim + 2\,p\sim^{3} l2\sim$$

$$+ 2\,p\sim l2\sim + 2\,p\sim^{4} l2\sim + 2\,l2\sim\big)\big)$$

```
> #il suit une loi normale, sa densite de probabilité :
  fROTd:=unapply((1/(sROTd*sqrt(2*Pi)))*exp(-(x-muROTd)^2/(2*sROTd^2)),x);
```

$fROTd := x$ **(12**

$$\rightarrow \frac{1}{2}\left(\sqrt{2}\right.$$

$$e^{-\frac{1}{2}\left(x - hops\sim\left(\frac{1}{2\,\lambda\sim}\left(2\,\lambda\sim Ed_ACK\sim + 2\,\lambda\sim DIFS\sim + 2\,\lambda\sim SIFS\sim + 2\,p\sim^{6} DIFS\sim\lambda\sim + 2\,p\sim^{5} ACK_Timeout\sim\lambda\sim + 2\,p\sim^{3} DIFS\sim\lambda\sim\right.\right.\right.}$$

$$+ 2\,p\sim^{3} ACK_Timeout\sim\lambda\sim + 4\,p\sim^{3} CWmin\sim\lambda\sim + 2\,p\sim^{7} DIFS\sim\lambda\sim + 64\,p\sim^{7} CWmin\sim\lambda\sim + 2\,p\sim^{7} ACK_Timeout\sim\lambda\sim + 2\,p\sim^{4} ACK_Timeout\sim\lambda\sim + 8\,p\sim^{4} CWmin\sim\lambda\sim$$

$$+ 2\,p\sim^{4} DIFS\sim\lambda\sim + 2\,p\sim ACK_Timeout\sim\lambda\sim + p\sim CWmin\sim\lambda\sim + 2\,p\sim DIFS\sim\lambda\sim + 2\,p\sim^{2} ACK_Timeout\sim\lambda\sim + 2\,p\sim^{2} CWmin\sim\lambda\sim + 2\,p\sim^{2} DIFS\sim\lambda\sim + 16\,p\sim^{5} CWmin\sim\lambda\sim$$

$$+ 2\, p\sim^{5}\, DIFS\sim\lambda\sim + 2\, p\sim^{6}\, ACK_Timeout\sim\lambda\sim + 32\, p\sim^{6}\, CWmin\sim\lambda\sim + 2\, p\sim l1\sim + 2\, p\sim^{4}\, l1\sim + 2\, p\sim^{2}\, l1\sim + 2\, p\sim^{6}\, l1\sim + 2\, p\sim^{5}\, l1\sim + 2\, p\sim^{3}\, l1\sim + 2\, p\sim^{7}\, l1\sim$$

$$+ 2\, l1\sim\Big) + \frac{1}{2\,\lambda\sim}\Big(2\,\lambda\sim Ed_ACK\sim + 2\,\lambda\sim DIFS\sim + 2\,\lambda\sim SIFS\sim + 2\, p\sim^{6}\, DIFS\sim\lambda\sim + 2\, p\sim^{5}\, ACK_Timeout\sim\lambda\sim + 2\, p\sim^{3}\, DIFS\sim\lambda\sim + 2\, p\sim^{3}\, ACK_Timeout\sim\lambda\sim$$

$$+ 4\, p\sim^{3}\, CWmin\sim\lambda\sim + 2\, p\sim^{7}\, DIFS\sim\lambda\sim + 64\, p\sim^{7}\, CWmin\sim\lambda\sim + 2\, p\sim^{7}\, ACK_Timeout\sim\lambda\sim + 2\, p\sim^{4}\, ACK_Timeout\sim\lambda\sim + 8\, p\sim^{4}\, CWmin\sim\lambda\sim + 2\, p\sim^{4}\, DIFS\sim\lambda\sim$$

$$+ 2\, p\sim ACK_Timeout\sim\lambda\sim + p\sim CWmin\sim\lambda\sim + 2\, p\sim DIFS\sim\lambda\sim + 2\, p\sim^{2}\, ACK_Timeout\sim\lambda\sim + 2\, p\sim^{2}\, CWmin\sim\lambda\sim + 2\, p\sim^{2}\, DIFS\sim\lambda\sim + 16\, p\sim^{5}\, CWmin\sim\lambda\sim + 2\, p\sim^{5}\, DIFS\sim\lambda\sim$$

$$+ 2\, p\sim^{6}\, ACK_Timeout\sim\lambda\sim + 32\, p\sim^{6}\, CWmin\sim\lambda\sim + 2\, p\sim^{6}\, l2\sim + 2\, p\sim^{2}\, l2\sim + 2\, p\sim^{7}\, l2\sim + 2\, p\sim^{5}\, l2\sim + 2\, p\sim^{3}\, l2\sim + 2\, p\sim l2\sim + 2\, p\sim^{4}\, l2\sim + 2\, l2\sim\Big)\Big)\Big)^{2} \Big/$$

$$\Bigg(hops\sim\Bigg(\frac{1}{4\,\lambda\sim^{2}}\Big(1020\, p\sim^{6}\, l1\sim CWmin\sim\lambda\sim + 104\, p\sim^{6}\, l1\sim DIFS\sim\lambda\sim + 188\, p\sim^{4}\, l1\sim CWmin\sim\lambda\sim + 72\, p\sim^{4}\, l1\sim DIFS\sim\lambda\sim + 64\, p\sim^{4}\, l1\sim ACK_Timeout\sim\lambda\sim$$

$$+ 8\, p\sim l1\sim SIFS\sim \lambda\sim + 8\, p\sim l1\sim Ed_ACK\sim \lambda\sim + 24\, p\sim l1\sim DIFS\sim \lambda\sim + 8\, p\sim^{7}\, l1\sim SIFS\sim \lambda\sim + 112\, p\sim^{7}\, l1\sim ACK_Timeout\sim \lambda\sim + 88\, p\sim^{5}\, l1\sim DIFS\sim \lambda\sim$$

$$+ 76\, p\sim^{3}\, l1\sim CWmin\sim \lambda\sim + 8\, p\sim^{3}\, l1\sim SIFS\sim \lambda\sim + 8\, p\sim^{3}\, l1\sim Ed_ACK\sim \lambda\sim + 8\, p\sim^{4}\, l1\sim SIFS\sim \lambda\sim + 8\, p\sim^{4}\, l1\sim Ed_ACK\sim \lambda\sim + 96\, p\sim^{6}\, l1\sim ACK_Timeout\sim \lambda\sim$$

$$+ 8\, p\sim^{6}\, l1\sim SIFS\sim \lambda\sim + 8\, p\sim^{5}\, l1\sim Ed_ACK\sim \lambda\sim + 8\, p\sim^{6}\, l1\sim Ed_ACK\sim \lambda\sim + 8\, p\sim^{7}\, l1\sim Ed_ACK\sim \lambda\sim + 48\, p\sim^{3}\, l1\sim ACK_Timeout\sim \lambda\sim + 8\, p\sim^{5}\, l1\sim SIFS\sim \lambda\sim$$

$$+ 56\, p\sim^{3}\, l1\sim DIFS\sim \lambda\sim + 444\, p\sim^{5}\, l1\sim CWmin\sim \lambda\sim + 4\, l1\sim^{2} + 28\, p\sim^{2}\, l1\sim CWmin\sim \lambda\sim + 8\, p\sim^{2}\, l1\sim SIFS\sim \lambda\sim + 8\, p\sim^{2}\, l1\sim Ed_ACK\sim \lambda\sim + 2300\, p\sim^{7}\, l1\sim CWmin\sim \lambda\sim$$

$$+ 120\, p\sim^{7}\, l1\sim DIFS\sim \lambda\sim + 8\, p\sim l1\sim CWmin\sim \lambda\sim + 80\, p\sim^{5}\, l1\sim ACK_Timeout\sim \lambda\sim + 8\, l1\sim \lambda\sim DIFS\sim + 8\, l1\sim \lambda\sim Ed_ACK\sim + 8\, l1\sim \lambda\sim SIFS\sim + 16\, p\sim l1\sim ACK_Timeout\sim \lambda\sim$$

$$+ 40\, p\sim^{2}\, l1\sim DIFS\sim \lambda\sim + 32\, p\sim^{2}\, l1\sim ACK_Timeout\sim \lambda\sim + 8\, p\sim^{3}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{3}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim^{3}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2}$$

$$+ 32\, p\sim^{4}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim SIFS\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 4\, p\sim SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 4\, p\sim ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim CWmin\sim DIFS\sim \lambda\sim^{2} + 64\, p\sim^{5}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 4\, p\sim Ed_ACK\sim CWmin\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{2}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 28\, p\sim^{2}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{2}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 32\, p\sim^{2}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 20\, p\sim^{2}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{2}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 2300\, p\sim^{7}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{7}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 256\, p\sim^{7}\, SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{7}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 256\, p\sim^{7}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 64\, p\sim^{4}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 112\, p\sim^{7}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2}$$

$$+ 2044\, p\sim^{7}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 156\, p\sim^{4}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 188\, p\sim^{4}\, CWmin\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{4}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 32\, p\sim^{4}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{4}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 8\, p\sim Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 444\, p\sim^{5}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 64\, p\sim^{5}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{6}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 128\, p\sim^{6}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 892\, p\sim^{6}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2}$$

$$+ 380\, p\sim^{5}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 96\, p\sim^{6}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 128\, p\sim^{6}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 1020\, p\sim^{6}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 48\, p\sim^{3}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim^{3}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{3}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 60\, p\sim^{3}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 76\, p\sim^{3}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{3}\, ACK_Timeout\sim\, SIFS\sim\, \lambda\sim^{2} + 80\, p\sim^{5}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2} + 4\, p\sim\, l1\sim + 4\, p\sim^{4}\, l1\sim$$

$$+ 4\, p\sim^{2}\, l1\sim + 36\, p\sim^{4}\, l1\sim^{2} + 12\, p\sim\, l1\sim^{2} + 20\, p\sim^{2}\, l1\sim^{2} + 28\, p\sim^{3}\, l1\sim^{2} + 60\, p\sim^{7}\, l1\sim^{2} + 44\, p\sim^{5}\, l1\sim^{2} + 52\, p\sim^{6}\, l1\sim^{2} + 4\, p\sim^{6}\, l1\sim + 4\, p\sim^{5}\, l1\sim + 4\, p\sim^{3}\, l1\sim$$

$$+ 20\, p\sim^{2}\, DIFS\sim^{2}\, \lambda\sim^{2} + 4\, p\sim^{7}\, l1\sim + 12\, p\sim\, DIFS\sim^{2}\, \lambda\sim^{2} + 36\, p\sim^{4}\, DIFS\sim^{2}\, \lambda\sim^{2} + 8\, p\sim^{2}\, CWmin\sim^{2}\, \lambda\sim^{2} + 60\, p\sim^{7}\, DIFS\sim^{2}\, \lambda\sim^{2} + 12160\, p\sim^{7}\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 52\, p\sim^{7}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 20\, p\sim^{3}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 40\, p\sim^{3}\, CWmin\sim^{2}\, \lambda\sim^{2} + 28\, p\sim^{3}\, DIFS\sim^{2}\, \lambda\sim^{2} + 736\, p\sim^{5}\, CWmin\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{5}\, DIFS\sim^{2}\, \lambda\sim^{2}$$

$$+ 36\, p\sim^{5}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 3008\, p\sim^{6}\, CWmin\sim^{2}\, \lambda\sim^{2} + 52\, p\sim^{6}\, DIFS\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{6}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 176\, p\sim^{4}\, CWmin\sim^{2}\, \lambda\sim^{2} + 28\, p\sim^{4}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2}$$

$$+ p\sim\, CWmin\sim^{2}\, \lambda\sim^{2} + 4\, p\sim\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 12\, p\sim^{2}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 8\, \lambda\sim^{2}\, DIFS\sim\, SIFS\sim + 8\, \lambda\sim^{2}\, Ed_ACK\sim\, SIFS\sim + 4\, \lambda\sim^{2}\, DIFS\sim^{2} + 4\, \lambda\sim^{2}\, Ed_ACK\sim^{2}$$

$$+ 4\,\lambda\sim^{2}\,SIFS\sim^{2} + 8\,\lambda\sim^{2}\,DIFS\sim\,Ed_ACK\sim + 4\,l1\sim) - \frac{1}{4\,\lambda\sim^{2}}(2\,\lambda\sim\,Ed_ACK\sim + 2\,\lambda\sim\,DIFS\sim + 2\,\lambda\sim\,SIFS\sim + 2\,p\sim^{6}\,DIFS\sim\lambda\sim + 2\,p\sim^{5}\,ACK_Timeout\sim\lambda\sim$$

$$+ 2\,p\sim^{3}\,DIFS\sim\lambda\sim + 2\,p\sim^{3}\,ACK_Timeout\sim\lambda\sim + 4\,p\sim^{3}\,CWmin\sim\lambda\sim + 2\,p\sim^{7}\,DIFS\sim\lambda\sim + 64\,p\sim^{7}\,CWmin\sim\lambda\sim + 2\,p\sim^{7}\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^{4}\,ACK_Timeout\sim\lambda\sim$$

$$+ 8\,p\sim^{4}\,CWmin\sim\lambda\sim + 2\,p\sim^{4}\,DIFS\sim\lambda\sim + 2\,p\sim\,ACK_Timeout\sim\lambda\sim + p\sim\,CWmin\sim\lambda\sim + 2\,p\sim\,DIFS\sim\lambda\sim + 2\,p\sim^{2}\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^{2}\,CWmin\sim\lambda\sim + 2\,p\sim^{2}\,DIFS\sim\lambda\sim$$

$$+ 16\,p\sim^{5}\,CWmin\sim\lambda\sim + 2\,p\sim^{5}\,DIFS\sim\lambda\sim + 2\,p\sim^{6}\,ACK_Timeout\sim\lambda\sim + 32\,p\sim^{6}\,CWmin\sim\lambda\sim + 2\,p\sim\,l1\sim + 2\,p\sim^{4}\,l1\sim + 2\,p\sim^{2}\,l1\sim + 2\,p\sim^{6}\,l1\sim + 2\,p\sim^{5}\,l1\sim + 2\,p\sim^{3}\,l1\sim$$

$$+ 2\,p\sim^{7}\,l1\sim + 2\,l1\sim)^{2} + \frac{1}{4\,\lambda\sim^{2}}(8\,p\sim\,l2\sim\,Ed_ACK\sim\lambda\sim + 24\,p\sim\,l2\sim\,DIFS\sim\lambda\sim + 8\,p\sim^{7}\,l2\sim\,SIFS\sim\lambda\sim + 112\,p\sim^{7}\,l2\sim\,ACK_Timeout\sim\lambda\sim + 104\,p\sim^{6}\,l2\sim\,DIFS\sim\lambda\sim$$

$$+ 188\,p\sim^{4}\,l2\sim\,CWmin\sim\lambda\sim + 72\,p\sim^{4}\,l2\sim\,DIFS\sim\lambda\sim + 64\,p\sim^{4}\,l2\sim\,ACK_Timeout\sim\lambda\sim + 8\,p\sim\,l2\sim\,SIFS\sim\lambda\sim + 76\,p\sim^{3}\,l2\sim\,CWmin\sim\lambda\sim + 8\,p\sim^{3}\,l2\sim\,SIFS\sim\lambda\sim$$

$$+ 8\, p\sim^{3}\, l2\sim Ed_ACK\sim\lambda\sim + 8\, p\sim^{4}\, l2\sim SIFS\sim\lambda\sim + 8\, p\sim^{4}\, l2\sim Ed_ACK\sim\lambda\sim + 96\, p\sim^{6}\, l2\sim ACK_Timeout\sim\lambda\sim + 8\, p\sim^{6}\, l2\sim SIFS\sim\lambda\sim + 1020\, p\sim^{6}\, l2\sim CWmin\sim\lambda\sim$$

$$+ 8\, p\sim^{5}\, l2\sim SIFS\sim\lambda\sim + 56\, p\sim^{3}\, l2\sim DIFS\sim\lambda\sim + 444\, p\sim^{5}\, l2\sim CWmin\sim\lambda\sim + 88\, p\sim^{5}\, l2\sim DIFS\sim\lambda\sim + 16\, p\sim l2\sim ACK_Timeout\sim\lambda\sim + 40\, p\sim^{2}\, l2\sim DIFS\sim\lambda\sim$$

$$+ 32\, p\sim^{2}\, l2\sim ACK_Timeout\sim\lambda\sim + 80\, p\sim^{5}\, l2\sim ACK_Timeout\sim\lambda\sim + 8\, p\sim^{5}\, l2\sim Ed_ACK\sim\lambda\sim + 8\, p\sim^{6}\, l2\sim Ed_ACK\sim\lambda\sim + 8\, p\sim^{7}\, l2\sim Ed_ACK\sim\lambda\sim$$

$$+ 48\, p\sim^{3}\, l2\sim ACK_Timeout\sim\lambda\sim + 2300\, p\sim^{7}\, l2\sim CWmin\sim\lambda\sim + 120\, p\sim^{7}\, l2\sim DIFS\sim\lambda\sim + 8\, p\sim l2\sim CWmin\sim\lambda\sim + 28\, p\sim^{3}\, l2\sim^{2} + 28\, p\sim^{2}\, l2\sim CWmin\sim\lambda\sim$$

$$+ 8\, p\sim^{2}\, l2\sim SIFS\sim\lambda\sim + 8\, p\sim^{2}\, l2\sim Ed_ACK\sim\lambda\sim + 4\, l2\sim^{2} + 8\, l2\sim\lambda\sim DIFS\sim + 8\, l2\sim\lambda\sim Ed_ACK\sim + 8\, l2\sim\lambda\sim SIFS\sim + 8\, p\sim^{3}\, SIFS\sim DIFS\sim\lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, Ed_ACK\sim DIFS\sim\lambda\sim^{2} + 16\, p\sim^{3}\, Ed_ACK\sim CWmin\sim\lambda\sim^{2} + 32\, p\sim^{4}\, SIFS\sim CWmin\sim\lambda\sim^{2} + 8\, p\sim SIFS\sim DIFS\sim\lambda\sim^{2} + 16\, p\sim ACK_Timeout\sim DIFS\sim\lambda\sim^{2}$$

$$+ 4\, p\sim SIFS\sim CWmin\sim \lambda\sim^2 + 8\, p\sim ACK_Timeout\sim Ed_ACK\sim \lambda\sim^2 + 4\, p\sim ACK_Timeout\sim CWmin\sim \lambda\sim^2 + 8\, p\sim ACK_Timeout\sim SIFS\sim \lambda\sim^2 + 8\, p\sim CWmin\sim DIFS\sim \lambda\sim^2$$

$$+ 64\, p\sim^5 SIFS\sim CWmin\sim \lambda\sim^2 + 4\, p\sim Ed_ACK\sim CWmin\sim \lambda\sim^2 + 8\, p\sim^2 SIFS\sim DIFS\sim \lambda\sim^2 + 8\, p\sim^2 Ed_ACK\sim CWmin\sim \lambda\sim^2 + 8\, p\sim^2 Ed_ACK\sim DIFS\sim \lambda\sim^2$$

$$+ 28\, p\sim^2 CWmin\sim DIFS\sim \lambda\sim^2 + 8\, p\sim^2 ACK_Timeout\sim Ed_ACK\sim \lambda\sim^2 + 8\, p\sim^2 ACK_Timeout\sim SIFS\sim \lambda\sim^2 + 32\, p\sim^2 ACK_Timeout\sim DIFS\sim \lambda\sim^2$$

$$+ 20\, p\sim^2 ACK_Timeout\sim CWmin\sim \lambda\sim^2 + 8\, p\sim^7 ACK_Timeout\sim Ed_ACK\sim \lambda\sim^2 + 8\, p\sim^2 SIFS\sim CWmin\sim \lambda\sim^2 + 2300\, p\sim^7 CWmin\sim DIFS\sim \lambda\sim^2$$

$$+ 8\, p\sim^7 ACK_Timeout\sim SIFS\sim \lambda\sim^2 + 8\, p\sim^7 Ed_ACK\sim DIFS\sim \lambda\sim^2 + 256\, p\sim^7 SIFS\sim CWmin\sim \lambda\sim^2 + 8\, p\sim^7 SIFS\sim DIFS\sim \lambda\sim^2 + 256\, p\sim^7 Ed_ACK\sim CWmin\sim \lambda\sim^2$$

$$+ 64\, p\sim^4 ACK_Timeout\sim DIFS\sim \lambda\sim^2 + 112\, p\sim^7 ACK_Timeout\sim DIFS\sim \lambda\sim^2 + 2044\, p\sim^7 ACK_Timeout\sim CWmin\sim \lambda\sim^2 + 8\, p\sim^4 ACK_Timeout\sim Ed_ACK\sim \lambda\sim^2$$

$$+ 156\, p\sim^{4}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 188\, p\sim^{4}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 32\, p\sim^{4}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{4}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 8\, p\sim Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 444\, p\sim^{5}\, CWmin\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{5}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 64\, p\sim^{5}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{6}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 128\, p\sim^{6}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 892\, p\sim^{6}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 380\, p\sim^{5}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 96\, p\sim^{6}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{6}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 128\, p\sim^{6}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 1020\, p\sim^{6}\, CWmin\sim DIFS\sim \lambda\sim^{2}$$

$$+ 48\, p\sim^{3}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim^{3}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{3}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 60\, p\sim^{3}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2}$$

$$+ 76\, p\sim^{3}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{3}\, ACK_Timeout\sim\, SIFS\sim\, \lambda\sim^{2} + 80\, p\sim^{5}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2} + 52\, p\sim^{6}\, l2\sim^{2} + 4\, p\sim^{6}\, l2\sim + 4\, p\sim^{2}\, l2\sim + 36\, p\sim^{4}\, l2\sim^{2}$$

$$+ 12\, p\sim\, l2\sim^{2} + 20\, p\sim^{2}\, l2\sim^{2} + 60\, p\sim^{7}\, l2\sim^{2} + 44\, p\sim^{5}\, l2\sim^{2} + 4\, p\sim^{7}\, l2\sim + 4\, p\sim^{5}\, l2\sim + 4\, p\sim^{3}\, l2\sim + 4\, p\sim\, l2\sim + 4\, p\sim^{4}\, l2\sim + 20\, p\sim^{2}\, DIFS\sim^{2}\, \lambda\sim^{2} + 12\, p\sim\, DIFS\sim^{2}\, \lambda\sim^{2}$$

$$+ 36\, p\sim^{4}\, DIFS\sim^{2}\, \lambda\sim^{2} + 8\, p\sim^{2}\, CWmin\sim^{2}\, \lambda\sim^{2} + 60\, p\sim^{7}\, DIFS\sim^{2}\, \lambda\sim^{2} + 12160\, p\sim^{7}\, CWmin\sim^{2}\, \lambda\sim^{2} + 52\, p\sim^{7}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 20\, p\sim^{3}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2}$$

$$+ 40\, p\sim^{3}\, CWmin\sim^{2}\, \lambda\sim^{2} + 28\, p\sim^{3}\, DIFS\sim^{2}\, \lambda\sim^{2} + 736\, p\sim^{5}\, CWmin\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{5}\, DIFS\sim^{2}\, \lambda\sim^{2} + 36\, p\sim^{5}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 3008\, p\sim^{6}\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 52\, p\sim^{6}\, DIFS\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{6}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 176\, p\sim^{4}\, CWmin\sim^{2}\, \lambda\sim^{2} + 28\, p\sim^{4}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + p\sim\, CWmin\sim^{2}\, \lambda\sim^{2} + 4\, p\sim\, ACK_Timeout\sim^{2}\, \lambda\sim^{2}$$

$$+ 12\, p\sim^{2}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 8\, \lambda\sim^{2}\, DIFS\sim\, SIFS\sim + 8\, \lambda\sim^{2}\, Ed_ACK\sim\, SIFS\sim + 4\, \lambda\sim^{2}\, DIFS\sim^{2} + 4\, \lambda\sim^{2}\, Ed_ACK\sim^{2} + 4\, \lambda\sim^{2}\, SIFS\sim^{2} + 8\, \lambda\sim^{2}\, DIFS\sim\, Ed_ACK\sim$$

$$+ 4\, l2\sim) - \frac{1}{4\,\lambda\sim^2}\big(2\,\lambda\sim Ed_ACK\sim + 2\,\lambda\sim DIFS\sim + 2\,\lambda\sim SIFS\sim + 2\,p\sim^6\, DIFS\sim\lambda\sim + 2\,p\sim^5\, ACK_Timeout\sim\lambda\sim + 2\,p\sim^3\, DIFS\sim\lambda\sim + 2\,p\sim^3\, ACK_Timeout\sim\lambda\sim$$

$$+ 4\,p\sim^3\, CWmin\sim\lambda\sim + 2\,p\sim^7\, DIFS\sim\lambda\sim + 64\,p\sim^7\, CWmin\sim\lambda\sim + 2\,p\sim^7\, ACK_Timeout\sim\lambda\sim + 2\,p\sim^4\, ACK_Timeout\sim\lambda\sim + 8\,p\sim^4\, CWmin\sim\lambda\sim + 2\,p\sim^4\, DIFS\sim\lambda\sim$$

$$+ 2\,p\sim ACK_Timeout\sim\lambda\sim + p\sim CWmin\sim\lambda\sim + 2\,p\sim DIFS\sim\lambda\sim + 2\,p\sim^2\, ACK_Timeout\sim\lambda\sim + 2\,p\sim^2\, CWmin\sim\lambda\sim + 2\,p\sim^2\, DIFS\sim\lambda\sim + 16\,p\sim^5\, CWmin\sim\lambda\sim + 2\,p\sim^5\, DIFS\sim\lambda\sim$$

$$+ 2\,p\sim^6\, ACK_Timeout\sim\lambda\sim + 32\,p\sim^6\, CWmin\sim\lambda\sim + 2\,p\sim^6\, l2\sim + 2\,p\sim^2\, l2\sim + 2\,p\sim^7\, l2\sim + 2\,p\sim^5\, l2\sim + 2\,p\sim^3\, l2\sim + 2\,p\sim l2\sim + 2\,p\sim^4\, l2\sim + 2\, l2\sim)^2)\Big)\Big) \Big/$$

$$\Big(\Big(hops\sim\Big(\frac{1}{4\,\lambda\sim^2}\big(1020\,p\sim^6\, l1\sim CWmin\sim\lambda\sim + 104\,p\sim^6\, l1\sim DIFS\sim\lambda\sim + 188\,p\sim^4\, l1\sim CWmin\sim\lambda\sim + 72\,p\sim^4\, l1\sim DIFS\sim\lambda\sim$$

$$+ 64\,p\sim^4\, l1\sim ACK_Timeout\sim\lambda\sim + 8\,p\sim l1\sim SIFS\sim\lambda\sim + 8\,p\sim l1\sim Ed_ACK\sim\lambda\sim + 24\,p\sim l1\sim DIFS\sim\lambda\sim + 8\,p\sim^7\, l1\sim SIFS\sim\lambda\sim$$

$$+ 112\,p\sim^7\, l1\sim ACK_Timeout\sim\lambda\sim + 88\,p\sim^5\, l1\sim DIFS\sim\lambda\sim + 76\,p\sim^3\, l1\sim CWmin\sim\lambda\sim + 8\,p\sim^3\, l1\sim SIFS\sim\lambda\sim + 8\,p\sim^3\, l1\sim Ed_ACK\sim\lambda\sim$$

$$+ 8\,p\sim^4\, l1\sim SIFS\sim\lambda\sim + 8\,p\sim^4\, l1\sim Ed_ACK\sim\lambda\sim + 96\,p\sim^6\, l1\sim ACK_Timeout\sim\lambda\sim + 8\,p\sim^6\, l1\sim SIFS\sim\lambda\sim + 8\,p\sim^5\, l1\sim Ed_ACK\sim\lambda\sim$$

$$+ 8\, p\sim^{6}\, l1\sim Ed_ACK\sim \lambda\sim + 8\, p\sim^{7}\, l1\sim Ed_ACK\sim \lambda\sim + 48\, p\sim^{3}\, l1\sim ACK_Timeout\sim \lambda\sim + 8\, p\sim^{5}\, l1\sim SIFS\sim \lambda\sim + 56\, p\sim^{3}\, l1\sim DIFS\sim \lambda\sim$$

$$+ 444\, p\sim^{5}\, l1\sim CWmin\sim \lambda\sim + 4\, l1\sim^{2} + 28\, p\sim^{2}\, l1\sim CWmin\sim \lambda\sim + 8\, p\sim^{2}\, l1\sim SIFS\sim \lambda\sim + 8\, p\sim^{2}\, l1\sim Ed_ACK\sim \lambda\sim$$

$$+ 2300\, p\sim^{7}\, l1\sim CWmin\sim \lambda\sim + 120\, p\sim^{7}\, l1\sim DIFS\sim \lambda\sim + 8\, p\sim l1\sim CWmin\sim \lambda\sim + 80\, p\sim^{5}\, l1\sim ACK_Timeout\sim \lambda\sim + 8\, l1\sim \lambda\sim DIFS\sim$$

$$+ 8\, l1\sim \lambda\sim Ed_ACK\sim + 8\, l1\sim \lambda\sim SIFS\sim + 16\, p\sim l1\sim ACK_Timeout\sim \lambda\sim + 40\, p\sim^{2}\, l1\sim DIFS\sim \lambda\sim + 32\, p\sim^{2}\, l1\sim ACK_Timeout\sim \lambda\sim$$

$$+ 8\, p\sim^{3}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{3}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim^{3}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 32\, p\sim^{4}\, SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 8\, p\sim SIFS\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 4\, p\sim SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 4\, p\sim ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim CWmin\sim DIFS\sim \lambda\sim^{2} + 64\, p\sim^{5}\, SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 4\, p\sim Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{2}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 28\, p\sim^{2}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2}$$

$$+ 32\, p\sim^{2}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 20\, p\sim^{2}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{2}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 2300\, p\sim^{7}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{7}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 256\, p\sim^{7}\, SIFS\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{7}\, SIFS\sim DIFS\sim \lambda\sim^{2} + 256\, p\sim^{7}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 64\, p\sim^{4}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2}$$

$$+ 112\, p\sim^{7}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 2044\, p\sim^{7}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2}$$

$$+ 156\, p\sim^{4}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 188\, p\sim^{4}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{4}\, SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 32\, p\sim^{4}\, Ed_ACK\sim CWmin\sim \lambda\sim^{2} + 8\, p\sim^{4}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 8\, p\sim Ed_ACK\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{5}\, Ed_ACK\sim DIFS\sim \lambda\sim^{2} + 444\, p\sim^{5}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{5}\, SIFS\sim DIFS\sim \lambda\sim^{2}$$

$$+ 64\, p\sim^{5}\, Ed_ACK\sim\, CWmin\sim\, \lambda\sim^{2} + 8\, p\sim^{6}\, SIFS\sim\, DIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{6}\, Ed_ACK\sim\, DIFS\sim\, \lambda\sim^{2} + 128\, p\sim^{6}\, Ed_ACK\sim\, CWmin\sim\, \lambda\sim^{2}$$

$$+ 892\, p\sim^{6}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 380\, p\sim^{5}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 96\, p\sim^{6}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 8\, p\sim^{6}\, ACK_Timeout\sim\, SIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{6}\, ACK_Timeout\sim\, Ed_ACK\sim\, \lambda\sim^{2} + 128\, p\sim^{6}\, SIFS\sim\, CWmin\sim\, \lambda\sim^{2}$$

$$+ 1020\, p\sim^{6}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2} + 48\, p\sim^{3}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2} + 16\, p\sim^{3}\, SIFS\sim\, CWmin\sim\, \lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, ACK_Timeout\sim\, Ed_ACK\sim\, \lambda\sim^{2} + 60\, p\sim^{3}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 76\, p\sim^{3}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, ACK_Timeout\sim\, SIFS\sim\, \lambda\sim^{2} + 80\, p\sim^{5}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2} + 4\, p\sim\, l1\sim + 4\, p\sim^{4}\, l1\sim + 4\, p\sim^{2}\, l1\sim + 36\, p\sim^{4}\, l1\sim^{2}$$

$$+ 12\, p\sim\, l1\sim^{2} + 20\, p\sim^{2}\, l1\sim^{2} + 28\, p\sim^{3}\, l1\sim^{2} + 60\, p\sim^{7}\, l1\sim^{2} + 44\, p\sim^{5}\, l1\sim^{2} + 52\, p\sim^{6}\, l1\sim^{2} + 4\, p\sim^{6}\, l1\sim + 4\, p\sim^{5}\, l1\sim + 4\, p\sim^{3}\, l1\sim$$

$$+ 20\, p\sim^{2}\, DIFS\sim^{2}\, \lambda\sim^{2} + 4\, p\sim^{7}\, l1\sim + 12\, p\sim\, DIFS\sim^{2}\, \lambda\sim^{2} + 36\, p\sim^{4}\, DIFS\sim^{2}\, \lambda\sim^{2} + 8\, p\sim^{2}\, CWmin\sim^{2}\, \lambda\sim^{2} + 60\, p\sim^{7}\, DIFS\sim^{2}\, \lambda\sim^{2}$$

$$+ 12160\, p\sim^{7}\, CWmin\sim^{2}\, \lambda\sim^{2} + 52\, p\sim^{7}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 20\, p\sim^{3}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 40\, p\sim^{3}\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 28\, p\sim^{3}\, DIFS\sim^{2}\, \lambda\sim^{2} + 736\, p\sim^{5}\, CWmin\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{5}\, DIFS\sim^{2}\, \lambda\sim^{2} + 36\, p\sim^{5}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 3008\, p\sim^{6}\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 52\, p\sim^{6}\, DIFS\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{6}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 176\, p\sim^{4}\, CWmin\sim^{2}\, \lambda\sim^{2} + 28\, p\sim^{4}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + p\sim\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 4\, p\sim\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 12\, p\sim^{2}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 8\, \lambda\sim^{2}\, DIFS\sim\, SIFS\sim + 8\, \lambda\sim^{2}\, Ed_ACK\sim\, SIFS\sim + 4\, \lambda\sim^{2}\, DIFS\sim^{2}$$

$$+ 4\, \lambda\sim^{2}\, Ed_ACK\sim^{2} + 4\, \lambda\sim^{2}\, SIFS\sim^{2} + 8\, \lambda\sim^{2}\, DIFS\sim\, Ed_ACK\sim + 4\, ll\sim\big) - \frac{1}{4\, \lambda\sim^{2}}\big(2\, \lambda\sim\, Ed_ACK\sim + 2\, \lambda\sim\, DIFS\sim + 2\, \lambda\sim\, SIFS\sim$$

$$+ 2\, p\sim^{6}\, DIFS\sim\, \lambda\sim + 2\, p\sim^{5}\, ACK_Timeout\sim\, \lambda\sim + 2\, p\sim^{3}\, DIFS\sim\, \lambda\sim + 2\, p\sim^{3}\, ACK_Timeout\sim\, \lambda\sim + 4\, p\sim^{3}\, CWmin\sim\, \lambda\sim + 2\, p\sim^{7}\, DIFS\sim\, \lambda\sim$$

$$+ 64\, p\sim^{7}\, CWmin\sim\, \lambda\sim + 2\, p\sim^{7}\, ACK_Timeout\sim\, \lambda\sim + 2\, p\sim^{4}\, ACK_Timeout\sim\, \lambda\sim + 8\, p\sim^{4}\, CWmin\sim\, \lambda\sim + 2\, p\sim^{4}\, DIFS\sim\, \lambda\sim$$

$$+ 2\, p\sim\, ACK_Timeout\sim\, \lambda\sim + p\sim\, CWmin\sim\, \lambda\sim + 2\, p\sim\, DIFS\sim\, \lambda\sim + 2\, p\sim^{2}\, ACK_Timeout\sim\, \lambda\sim + 2\, p\sim^{2}\, CWmin\sim\, \lambda\sim + 2\, p\sim^{2}\, DIFS\sim\, \lambda\sim$$

$$+ 16\, p\sim^{5}\, CWmin\sim\lambda\sim + 2\, p\sim^{5}\, DIFS\sim\lambda\sim + 2\, p\sim^{6}\, ACK_Timeout\sim\lambda\sim + 32\, p\sim^{6}\, CWmin\sim\lambda\sim + 2\, p\sim l1\sim + 2\, p\sim^{4}\, l1\sim + 2\, p\sim^{2}\, l1\sim$$

$$+ 2\, p\sim^{6}\, l1\sim + 2\, p\sim^{5}\, l1\sim + 2\, p\sim^{3}\, l1\sim + 2\, p\sim^{7}\, l1\sim + 2\, l1\sim\big)^{2} + \frac{1}{4\,\lambda\sim^{2}}\big(8\, p\sim l2\sim Ed_ACK\sim\lambda\sim + 24\, p\sim l2\sim DIFS\sim\lambda\sim$$

$$+ 8\, p\sim^{7}\, l2\sim SIFS\sim\lambda\sim + 112\, p\sim^{7}\, l2\sim ACK_Timeout\sim\lambda\sim + 104\, p\sim^{6}\, l2\sim DIFS\sim\lambda\sim + 188\, p\sim^{4}\, l2\sim CWmin\sim\lambda\sim + 72\, p\sim^{4}\, l2\sim DIFS\sim\lambda\sim$$

$$+ 64\, p\sim^{4}\, l2\sim ACK_Timeout\sim\lambda\sim + 8\, p\sim l2\sim SIFS\sim\lambda\sim + 76\, p\sim^{3}\, l2\sim CWmin\sim\lambda\sim + 8\, p\sim^{3}\, l2\sim SIFS\sim\lambda\sim + 8\, p\sim^{3}\, l2\sim Ed_ACK\sim\lambda\sim$$

$$+ 8\, p\sim^{4}\, l2\sim SIFS\sim\lambda\sim + 8\, p\sim^{4}\, l2\sim Ed_ACK\sim\lambda\sim + 96\, p\sim^{6}\, l2\sim ACK_Timeout\sim\lambda\sim + 8\, p\sim^{6}\, l2\sim SIFS\sim\lambda\sim + 1020\, p\sim^{6}\, l2\sim CWmin\sim\lambda\sim$$

$$+ 8\, p\sim^{5}\, l2\sim SIFS\sim\lambda\sim + 56\, p\sim^{3}\, l2\sim DIFS\sim\lambda\sim + 444\, p\sim^{5}\, l2\sim CWmin\sim\lambda\sim + 88\, p\sim^{5}\, l2\sim DIFS\sim\lambda\sim + 16\, p\sim l2\sim ACK_Timeout\sim\lambda\sim$$

$$+ 40\, p\sim^{2}\, l2\sim DIFS\sim\lambda\sim + 32\, p\sim^{2}\, l2\sim ACK_Timeout\sim\lambda\sim + 80\, p\sim^{5}\, l2\sim ACK_Timeout\sim\lambda\sim + 8\, p\sim^{5}\, l2\sim Ed_ACK\sim\lambda\sim$$

$$+ 8\, p\sim^{6}\, l2\sim Ed_ACK\sim\lambda\sim + 8\, p\sim^{7}\, l2\sim Ed_ACK\sim\lambda\sim + 48\, p\sim^{3}\, l2\sim ACK_Timeout\sim\lambda\sim + 2300\, p\sim^{7}\, l2\sim CWmin\sim\lambda\sim$$

$$+ 120\, p\sim^{7}\, l2\sim DIFS\sim\lambda\sim + 8\, p\sim l2\sim CWmin\sim\lambda\sim + 28\, p\sim^{3}\, l2\sim^{2} + 28\, p\sim^{2}\, l2\sim CWmin\sim\lambda\sim + 8\, p\sim^{2}\, l2\sim SIFS\sim\lambda\sim$$

$$+ 8\, p\sim^{2}\, l2\sim Ed_ACK\sim\lambda\sim + 4\, l2\sim^{2} + 8\, l2\sim\lambda\sim DIFS\sim + 8\, l2\sim\lambda\sim Ed_ACK\sim + 8\, l2\sim\lambda\sim SIFS\sim + 8\, p\sim^{3}\, SIFS\sim DIFS\sim\lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, Ed_ACK\sim DIFS\sim\lambda\sim^{2} + 16\, p\sim^{3}\, Ed_ACK\sim CWmin\sim\lambda\sim^{2} + 32\, p\sim^{4}\, SIFS\sim CWmin\sim\lambda\sim^{2} + 8\, p\sim SIFS\sim DIFS\sim\lambda\sim^{2}$$

$$+ 16\, p\sim ACK_Timeout\sim DIFS\sim\lambda\sim^{2} + 4\, p\sim SIFS\sim CWmin\sim\lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim Ed_ACK\sim\lambda\sim^{2}$$

$$+ 4\, p\sim ACK_Timeout\sim CWmin\sim\lambda\sim^{2} + 8\, p\sim ACK_Timeout\sim SIFS\sim\lambda\sim^{2} + 8\, p\sim CWmin\sim DIFS\sim\lambda\sim^{2} + 64\, p\sim^{5}\, SIFS\sim CWmin\sim\lambda\sim^{2}$$

$$+ 4\, p\sim Ed_ACK\sim CWmin\sim\lambda\sim^{2} + 8\, p\sim^{2}\, SIFS\sim DIFS\sim\lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim CWmin\sim\lambda\sim^{2} + 8\, p\sim^{2}\, Ed_ACK\sim DIFS\sim\lambda\sim^{2}$$

$$+ 28\, p\sim^{2}\, CWmin\sim DIFS\sim\lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim Ed_ACK\sim\lambda\sim^{2} + 8\, p\sim^{2}\, ACK_Timeout\sim SIFS\sim\lambda\sim^{2}$$

$$+ 32\, p\sim^{2}\, ACK_Timeout\sim DIFS\sim\lambda\sim^{2} + 20\, p\sim^{2}\, ACK_Timeout\sim CWmin\sim\lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim Ed_ACK\sim\lambda\sim^{2}$$

$$+ 8\, p\sim^{2}\, SIFS\sim\, CWmin\sim\, \lambda\sim^{2} + 2300\, p\sim^{7}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{7}\, ACK_Timeout\sim\, SIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{7}\, Ed_ACK\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 256\, p\sim^{7}\, SIFS\sim\, CWmin\sim\, \lambda\sim^{2} + 8\, p\sim^{7}\, SIFS\sim\, DIFS\sim\, \lambda\sim^{2} + 256\, p\sim^{7}\, Ed_ACK\sim\, CWmin\sim\, \lambda\sim^{2} + 64\, p\sim^{4}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 112\, p\sim^{7}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2} + 2044\, p\sim^{7}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim\, Ed_ACK\sim\, \lambda\sim^{2}$$

$$+ 156\, p\sim^{4}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 188\, p\sim^{4}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{4}\, ACK_Timeout\sim\, SIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{4}\, SIFS\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 32\, p\sim^{4}\, Ed_ACK\sim\, CWmin\sim\, \lambda\sim^{2} + 8\, p\sim^{4}\, Ed_ACK\sim\, DIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim\, Ed_ACK\sim\, \lambda\sim^{2} + 8\, p\sim\, Ed_ACK\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 8\, p\sim^{5}\, Ed_ACK\sim\, DIFS\sim\, \lambda\sim^{2} + 444\, p\sim^{5}\, CWmin\sim\, DIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{5}\, ACK_Timeout\sim\, SIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{5}\, SIFS\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 64\, p\sim^{5}\, Ed_ACK\sim\, CWmin\sim\, \lambda\sim^{2} + 8\, p\sim^{6}\, SIFS\sim\, DIFS\sim\, \lambda\sim^{2} + 8\, p\sim^{6}\, Ed_ACK\sim\, DIFS\sim\, \lambda\sim^{2} + 128\, p\sim^{6}\, Ed_ACK\sim\, CWmin\sim\, \lambda\sim^{2}$$

$$+ 892\, p\sim^{6}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 380\, p\sim^{5}\, ACK_Timeout\sim\, CWmin\sim\, \lambda\sim^{2} + 96\, p\sim^{6}\, ACK_Timeout\sim\, DIFS\sim\, \lambda\sim^{2}$$

$$+ 8\, p\sim^{6}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 8\, p\sim^{6}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 128\, p\sim^{6}\, SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 1020\, p\sim^{6}\, CWmin\sim DIFS\sim \lambda\sim^{2} + 48\, p\sim^{3}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 16\, p\sim^{3}\, SIFS\sim CWmin\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, ACK_Timeout\sim Ed_ACK\sim \lambda\sim^{2} + 60\, p\sim^{3}\, ACK_Timeout\sim CWmin\sim \lambda\sim^{2} + 76\, p\sim^{3}\, CWmin\sim DIFS\sim \lambda\sim^{2}$$

$$+ 8\, p\sim^{3}\, ACK_Timeout\sim SIFS\sim \lambda\sim^{2} + 80\, p\sim^{5}\, ACK_Timeout\sim DIFS\sim \lambda\sim^{2} + 52\, p\sim^{6}\, l2\sim^{2} + 4\, p\sim^{6}\, l2\sim + 4\, p\sim^{2}\, l2\sim + 36\, p\sim^{4}\, l2\sim^{2}$$

$$+ 12\, p\sim l2\sim^{2} + 20\, p\sim^{2}\, l2\sim^{2} + 60\, p\sim^{7}\, l2\sim^{2} + 44\, p\sim^{5}\, l2\sim^{2} + 4\, p\sim^{7}\, l2\sim + 4\, p\sim^{5}\, l2\sim + 4\, p\sim^{3}\, l2\sim + 4\, p\sim l2\sim + 4\, p\sim^{4}\, l2\sim$$

$$+ 20\, p\sim^{2}\, DIFS\sim^{2}\, \lambda\sim^{2} + 12\, p\sim DIFS\sim^{2}\, \lambda\sim^{2} + 36\, p\sim^{4}\, DIFS\sim^{2}\, \lambda\sim^{2} + 8\, p\sim^{2}\, CWmin\sim^{2}\, \lambda\sim^{2} + 60\, p\sim^{7}\, DIFS\sim^{2}\, \lambda\sim^{2}$$

$$+ 12160\, p\sim^{7}\, CWmin\sim^{2}\, \lambda\sim^{2} + 52\, p\sim^{7}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 20\, p\sim^{3}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 40\, p\sim^{3}\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 28\, p\sim^{3}\, DIFS\sim^{2}\, \lambda\sim^{2} + 736\, p\sim^{5}\, CWmin\sim^{2}\, \lambda\sim^{2} + 44\, p\sim^{5}\, DIFS\sim^{2}\, \lambda\sim^{2} + 36\, p\sim^{5}\, ACK_Timeout\sim^{2}\, \lambda\sim^{2} + 3008\, p\sim^{6}\, CWmin\sim^{2}\, \lambda\sim^{2}$$

$$+ 52\, p\sim^6 DIFS\sim^2 \lambda\sim^2 + 44\, p\sim^6 ACK_Timeout\sim^2 \lambda\sim^2 + 176\, p\sim^4 CWmin\sim^2 \lambda\sim^2 + 28\, p\sim^4 ACK_Timeout\sim^2 \lambda\sim^2 + p\sim CWmin\sim^2 \lambda\sim^2$$

$$+ 4\, p\sim ACK_Timeout\sim^2 \lambda\sim^2 + 12\, p\sim^2 ACK_Timeout\sim^2 \lambda\sim^2 + 8\, \lambda\sim^2 DIFS\sim SIFS\sim + 8\, \lambda\sim^2 Ed_ACK\sim SIFS\sim + 4\, \lambda\sim^2 DIFS\sim^2$$

$$+ 4\, \lambda\sim^2 Ed_ACK\sim^2 + 4\, \lambda\sim^2 SIFS\sim^2 + 8\, \lambda\sim^2 DIFS\sim Ed_ACK\sim + 4\, l2\sim\big) - \frac{1}{4\, \lambda\sim^2}\big(2\, \lambda\sim Ed_ACK\sim + 2\, \lambda\sim DIFS\sim + 2\, \lambda\sim SIFS\sim$$

$$+ 2\, p\sim^6 DIFS\sim \lambda\sim + 2\, p\sim^5 ACK_Timeout\sim \lambda\sim + 2\, p\sim^3 DIFS\sim \lambda\sim + 2\, p\sim^3 ACK_Timeout\sim \lambda\sim + 4\, p\sim^3 CWmin\sim \lambda\sim + 2\, p\sim^7 DIFS\sim \lambda\sim$$

$$+ 64\, p\sim^7 CWmin\sim \lambda\sim + 2\, p\sim^7 ACK_Timeout\sim \lambda\sim + 2\, p\sim^4 ACK_Timeout\sim \lambda\sim + 8\, p\sim^4 CWmin\sim \lambda\sim + 2\, p\sim^4 DIFS\sim \lambda\sim$$

$$+ 2\, p\sim ACK_Timeout\sim \lambda\sim + p\sim CWmin\sim \lambda\sim + 2\, p\sim DIFS\sim \lambda\sim + 2\, p\sim^2 ACK_Timeout\sim \lambda\sim + 2\, p\sim^2 CWmin\sim \lambda\sim + 2\, p\sim^2 DIFS\sim \lambda\sim$$

$$+ 16\, p\sim^5 CWmin\sim \lambda\sim + 2\, p\sim^5 DIFS\sim \lambda\sim + 2\, p\sim^6 ACK_Timeout\sim \lambda\sim + 32\, p\sim^6 CWmin\sim \lambda\sim + 2\, p\sim^6 l2\sim + 2\, p\sim^2 l2\sim + 2\, p\sim^7 l2\sim$$

$$+ 2\, p\sim^5 l2\sim + 2\, p\sim^3 l2\sim + 2\, p\sim l2\sim + 2\, p\sim^4 l2\sim + 2\, l2\sim\big)^2\big)\big)^{1/2} \sqrt{\pi}\Big)$$

```
> #valeurs des paramètres :
  #pour la 1ère phase :
  Valeurs:=[DIFS=10.0E-06+2*20.0E-06, CWmin=32*20.0E-06, SIFS=10.0E-06, ACK_Timeout=334.0E-06,
  Ed_ACK=304.0E-06, l1=287, l2=32, lambda=(11.0E06/8), p=0.1];
> #pour la 2ème phase :
  Valeurs0:=[DIFS=10.0E-06+2*20.0E-06, CWmin=32*20.0E-06, SIFS=10.0E-06, ACK_Timeout=334.0E-06,
  Ed_ACK=304.0E-06, l1=1593, l2=1925, lambda=(11.0E06/8), p=0.1];
```

$Valeurs := [DIFS\sim = 0.0000500,\ CWmin\sim = 0.0006400,\ SIFS\sim = 0.0000100,\ ACK_Timeout\sim = 0.0003340,\ Ed_ACK\sim = 0.0003040,\ l1\sim = 287,$ **(13**

$l2\sim = 32, \lambda\sim = 1.375000000\ 10^{6}, p\sim = 0.1]$

$Valeurs0 := [DIFS\sim = 0.0000500, CWmin\sim = 0.0006400, SIFS\sim = 0.0000100, ACK_Timeout\sim = 0.0003340, Ed_ACK\sim = 0.0003040, l1\sim = 1593, l2\sim = 1925, \lambda\sim = 1.375000000\ 10^{6}, p\sim = 0.1]$

```
> #temps d'un aller-retour à la 1ère phase, sa densité de probabilité :
  fROTd_1:=unapply(eval(subs(Valeurs, fROTd(x))),x);
> #temps d'un aller-retour à la 2ème phase, sa densité de probabilité :
  fROTd_2:=unapply(eval(subs(Valeurs0, fROTd(x))),x);
```

$$fROTd_1 := x \rightarrow \frac{1086.283072\ \sqrt{2}\ e^{-\frac{2.360021823\ 10^{6}\ (x - 0.001151110076\ hops\sim)^{2}}{hops\sim}}}{\sqrt{hops\sim}\ \sqrt{\pi}} \quad (14$$

$$fROTd_2 := x \rightarrow \frac{483.5485222\ \sqrt{2}\ e^{-\frac{4.676383466\ 10^{5}\ (x - 0.003736160555\ hops\sim)^{2}}{hops\sim}}}{\sqrt{hops\sim}\ \sqrt{\pi}}$$

```
> #temps d'un aller-retour à la 1ère phase, sa fonction de répartition :
  int(fROTd_1(x),x=-infinity..infinity);
> #temps d'un aller-retour à la 2ème phase, sa fonction de répartition :
  int(fROTd_2(x),x=-infinity..infinity);
```

$$1.000000000 \quad (15$$

$$1.000000000$$

```
> #temps d'aller-retours global à la 1ère phase, sa fonction de répartition :
  FROTd_max_1:=simplify(int(fROTd_1(x),x=-infinity..t)^n);
> #temps d'aller-retours global à la 2ème phase, sa fonction de répartition :
  FROTd_max_2:=simplify(int(fROTd_2(x),x=-infinity..t)^n);
```

$$FROTd_max_1 := \left(-0.5000000002\ \mathrm{erf}\left(\frac{6.144945009\ 10^{-9}\ (-2.500000000\ 10^{11}\ t\sim + 2.87777519\ 10^{8}\ hops\sim)}{\sqrt{hops\sim}}\right) + 0.5000000002\right)^{n\sim} \quad (16$$

$$FROTd_max_2 := \left(-0.5000000000\ \mathrm{erf}\left(\frac{3.419204391\ 10^{-9}\ (-2.000000000\ 10^{11}\ t\sim + 7.47232111\ 10^{8}\ hops\sim)}{\sqrt{hops\sim}}\right) + 0.5000000000\right)^{n\sim}$$

```
> #temps d'aller-retours global à la 1ère phase, sa densité de probabilité :
```

```
fROTd_max_1:=diff(FROTd_max_1,t);
> #temps d'aller-retours global à la 2ème phase, sa densité de probabilité :
fROTd_max_2:=diff(FROTd_max_2,t);
```

$$fROTd_max_1 := \left(1536.236253\left(-0.5000000002\,\mathrm{erf}\left(\frac{6.144945009\,10^{-9}\left(-2.500000000\,10^{11}\,t\sim+2.87777519\,10^{8}\,hops\sim\right)}{\sqrt{hops\sim}}\right)+0.5000000002\right)^{n\sim} n\sim \mathrm{e}^{-\frac{3.776034916\,10^{-17}\left(-2.500000000\,10^{11}\,t\sim+2.87777519\,10^{8}\,hops\sim\right)^{2}}{hops\sim}}\right) \Big/ \left(\sqrt{\pi}\,\sqrt{hops\sim}\left(-0.5000000002\,\mathrm{erf}\left(\frac{6.144945009\,10^{-9}\left(-2.500000000\,10^{11}\,t\sim+2.87777519\,10^{8}\,hops\sim\right)}{\sqrt{hops\sim}}\right)+0.5000000002\right)\right) \quad (17$$

$$fROTd_max_2 := \left(683.8408780\left(-0.5000000000\,\mathrm{erf}\left(\frac{3.419204391\,10^{-9}\left(-2.000000000\,10^{11}\,t\sim+7.47232111\,10^{8}\,hops\sim\right)}{\sqrt{hops\sim}}\right)+0.5000000000\right)^{n\sim} n\sim \mathrm{e}^{-\frac{1.169095867\,10^{-17}\left(-2.000000000\,10^{11}\,t\sim+7.47232111\,10^{8}\,hops\sim\right)^{2}}{hops\sim}}\right) \Big/ \left(\sqrt{\pi}\,\sqrt{hops\sim}\left(-0.5000000000\,\mathrm{erf}\left(\frac{3.419204391\,10^{-9}\left(-2.000000000\,10^{11}\,t\sim+7.47232111\,10^{8}\,hops\sim\right)}{\sqrt{hops\sim}}\right)+0.5000000000\right)\right)$$

```
> #temps d'aller-retours global à la 1ère phase, sa moyenne :
phi1:=eval(simplify(int(t*fROTd_max_1,t=-infinity..infinity)));
> #temps d'aller-retours global à la 2ème phase, sa moyenne :
phi2:=eval(simplify(int(t*fROTd_max_2,t=-infinity..infinity)));
```

$$\phi 1 := \int_{-Float(\infty)}^{Float(\infty)} \frac{1}{\sqrt{hops\sim}}\left(866.7284917\,t\sim\left(-0.5000000002\,\mathrm{erf}\left(\frac{6.144945009\,10^{-9}\left(-2.500000000\,10^{11}\,t\sim+2.87777519\,10^{8}\,hops\sim\right)}{\sqrt{hops\sim}}\right)\right.\right. \quad (18$$

$$+ 0.5000000002\Bigg)^{n\sim - 1.} n\sim \mathrm{e}^{-\frac{3.776034916\ 10^{-17}\left(-2.500000000\ 10^{11}\ t\sim + 2.87777519\ 10^{8}\ hops\sim\right)^{2}}{hops\sim}}\Bigg)\,\mathrm{d}t\sim$$

$$\phi 2 := \int_{-Float(\infty)}^{Float(\infty)} \frac{1}{\sqrt{hops\sim}}\Bigg(385.8159001\ t\sim\Bigg(-0.5000000000\ \mathrm{erf}\Bigg(\frac{3.419204391\ 10^{-9}\left(-2.000000000\ 10^{11}\ t\sim + 7.47232111\ 10^{8}\ hops\sim\right)}{\sqrt{hops\sim}}\Bigg)$$

$$+ 0.5000000000\Bigg)^{n\sim - 1.} n\sim \mathrm{e}^{-\frac{1.169095867\ 10^{-17}\left(-2.000000000\ 10^{11}\ t\sim + 7.47232111\ 10^{8}\ hops\sim\right)^{2}}{hops\sim}}\Bigg)\,\mathrm{d}t\sim$$

```
> #temps d'authentification, sa moyenne :
  phi:=eval(simplify(phi1+phi2));
  x:=evalf(subs([hops=4,n=5],phi1));
  y:=evalf(subs([hops=4,n=5],phi2));
  z:=evalf(x+y);
```

$$x := 1.427400810\ 10^{-13} \quad \textbf{(19}$$

$$y := 0.01734970482$$

$$z := 0.01734970482$$

```
> x:=evalf(subs([hops=4,n=6],phi1));
  y:=evalf(subs([hops=4,n=6],phi2));
  z:=evalf(x+y);
```

$$x := 6.389872858\ 10^{-16} \quad \textbf{(20}$$

$$y := 0.01756528171$$

$$z := 0.01756528171$$

```
> for i from 1 by 1 to 10 do
   for j from 1 by 1 to 6 do
   x:=evalf(subs([hops=i,n=j],phi1));
   y:=evalf(subs([hops=i,n=j],phi2));
   #valeur de la moyenne du temps d'authentification en (hops=i,n=j) :
   z:=evalf(x+y);
    printf("%d    %g\n",j,z);
```

## E.2 Modèle somme

```
  #sur un lien à sauts multiples de longueur hops :
  #temps d'un aller-retour à la 1ère phase, sa moyenne :
  muROTd_1:=hops*(muTRd_k(l1)+muTRd_k(l2));
> #temps d'un aller-retour à la 2ème phase, sa moyenne :
  muROTd_2:=hops*(muTRd_k(l3)+muTRd_k(l4));
```

$$muROTd_1 := hops\sim \left( \frac{1}{2\,\lambda\sim} \left(2\,p\sim^3\,ACK_Timeout\sim\lambda\sim + 2\,p\sim\,ACK_Timeout\sim\lambda\sim + p\sim\,CWmin\sim\lambda\sim + 2\,p\sim\,DIFS\sim\lambda\sim \right.\right.$$

$$+ 2\,p\sim^5\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^7\,ACK_Timeout\sim\lambda\sim + 64\,p\sim^7\,CWmin\sim\lambda\sim + 2\,p\sim^4\,ACK_Timeout\sim\lambda\sim + 8\,p\sim^4\,CWmin\sim\lambda\sim$$

$$+ 2\,p\sim^4\,DIFS\sim\lambda\sim + 16\,p\sim^5\,CWmin\sim\lambda\sim + 2\,p\sim^5\,DIFS\sim\lambda\sim + 2\,p\sim^2\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^2\,CWmin\sim\lambda\sim + 2\,p\sim^2\,DIFS\sim\lambda\sim$$

$$+ 2\,p\sim^6\,DIFS\sim\lambda\sim + 2\,p\sim^6\,ACK_Timeout\sim\lambda\sim + 32\,p\sim^6\,CWmin\sim\lambda\sim + 4\,p\sim^3\,CWmin\sim\lambda\sim + 2\,p\sim^3\,DIFS\sim\lambda\sim + 2\,p\sim^7\,DIFS\sim\lambda\sim$$

$$+ 2\,\lambda\sim SIFS\sim + 2\,l1\sim + 2\,\lambda\sim DIFS\sim + 2\,\lambda\sim Ed_ACK\sim + 2\,p\sim^2\,l1\sim + 2\,p\sim^5\,l1\sim + 2\,p\sim^7\,l1\sim + 2\,p\sim^3\,l1\sim + 2\,p\sim^4\,l1\sim + 2\,p\sim^6\,l1\sim$$

$$\left. + 2\,p\sim\,l1\sim \right) + \frac{1}{2\,\lambda\sim} \left(2\,p\sim^3\,ACK_Timeout\sim\lambda\sim + 2\,p\sim\,ACK_Timeout\sim\lambda\sim + p\sim\,CWmin\sim\lambda\sim + 2\,p\sim\,DIFS\sim\lambda\sim \right.$$

$$+ 2\,p\sim^5\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^7\,ACK_Timeout\sim\lambda\sim + 64\,p\sim^7\,CWmin\sim\lambda\sim + 2\,p\sim^4\,ACK_Timeout\sim\lambda\sim + 8\,p\sim^4\,CWmin\sim\lambda\sim$$

$$+ 2\,p\sim^4\,DIFS\sim\lambda\sim + 16\,p\sim^5\,CWmin\sim\lambda\sim + 2\,p\sim^5\,DIFS\sim\lambda\sim + 2\,p\sim^2\,ACK_Timeout\sim\lambda\sim + 2\,p\sim^2\,CWmin\sim\lambda\sim + 2\,p\sim^2\,DIFS\sim\lambda\sim$$

$$+ 2\,p\sim^6\,DIFS\sim\lambda\sim + 2\,p\sim^6\,ACK_Timeout\sim\lambda\sim + 32\,p\sim^6\,CWmin\sim\lambda\sim + 4\,p\sim^3\,CWmin\sim\lambda\sim + 2\,p\sim^3\,DIFS\sim\lambda\sim + 2\,p\sim^7\,DIFS\sim\lambda\sim$$

$$+ 2\,\lambda\sim SIFS\sim + 2\,l2\sim + 2\,\lambda\sim DIFS\sim + 2\,\lambda\sim Ed_ACK\sim + 2\,p\sim^4\,l2\sim + 2\,p\sim\,l2\sim + 2\,p\sim^6\,l2\sim + 2\,p\sim^5\,l2\sim + 2\,p\sim^7\,l2\sim + 2\,p\sim^3\,l2\sim$$

$$\left. \left. + 2\,p\sim^2\,l2\sim \right) \right) \quad (9$$

$$muROTd_2 := hops\sim \left( \frac{1}{2\,\lambda\sim} \left(2\,p\sim^3\,ACK_Timeout\sim\lambda\sim + 2\,p\sim\,ACK_Timeout\sim\lambda\sim + p\sim\,CWmin\sim\lambda\sim + 2\,p\sim\,DIFS\sim\lambda\sim \right.\right.$$

$$+ 2\, p\sim^{5}\, ACK_Timeout\sim \lambda\sim + 2\, p\sim^{7}\, ACK_Timeout\sim \lambda\sim + 64\, p\sim^{7}\, CWmin\sim \lambda\sim + 2\, p\sim^{4}\, ACK_Timeout\sim \lambda\sim + 8\, p\sim^{4}\, CWmin\sim \lambda\sim$$

$$+ 2\, p\sim^{4}\, DIFS\sim \lambda\sim + 16\, p\sim^{5}\, CWmin\sim \lambda\sim + 2\, p\sim^{5}\, DIFS\sim \lambda\sim + 2\, p\sim^{2}\, ACK_Timeout\sim \lambda\sim + 2\, p\sim^{2}\, CWmin\sim \lambda\sim + 2\, p\sim^{2}\, DIFS\sim \lambda\sim$$

$$+ 2\, p\sim^{6}\, DIFS\sim \lambda\sim + 2\, p\sim^{6}\, ACK_Timeout\sim \lambda\sim + 32\, p\sim^{6}\, CWmin\sim \lambda\sim + 4\, p\sim^{3}\, CWmin\sim \lambda\sim + 2\, p\sim^{3}\, DIFS\sim \lambda\sim + 2\, p\sim^{7}\, DIFS\sim \lambda\sim$$

$$+ 2\, \lambda\sim SIFS\sim + 2\, l3\sim + 2\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim + 2\, p\sim^{3}\, l3\sim + 2\, p\sim^{6}\, l3\sim + 2\, p\sim^{4}\, l3\sim + 2\, p\sim^{2}\, l3\sim + 2\, p\sim^{5}\, l3\sim + 2\, p\sim^{7}\, l3\sim$$

$$+ 2\, p\sim l3\sim) + \frac{1}{2\, \lambda\sim}\left(2\, p\sim^{3}\, ACK_Timeout\sim \lambda\sim + 2\, p\sim ACK_Timeout\sim \lambda\sim + p\sim CWmin\sim \lambda\sim + 2\, p\sim DIFS\sim \lambda\sim\right.$$

$$+ 2\, p\sim^{5}\, ACK_Timeout\sim \lambda\sim + 2\, p\sim^{7}\, ACK_Timeout\sim \lambda\sim + 64\, p\sim^{7}\, CWmin\sim \lambda\sim + 2\, p\sim^{4}\, ACK_Timeout\sim \lambda\sim + 8\, p\sim^{4}\, CWmin\sim \lambda\sim$$

$$+ 2\, p\sim^{4}\, DIFS\sim \lambda\sim + 16\, p\sim^{5}\, CWmin\sim \lambda\sim + 2\, p\sim^{5}\, DIFS\sim \lambda\sim + 2\, p\sim^{2}\, ACK_Timeout\sim \lambda\sim + 2\, p\sim^{2}\, CWmin\sim \lambda\sim + 2\, p\sim^{2}\, DIFS\sim \lambda\sim$$

$$+ 2\, p\sim^{6}\, DIFS\sim \lambda\sim + 2\, p\sim^{6}\, ACK_Timeout\sim \lambda\sim + 32\, p\sim^{6}\, CWmin\sim \lambda\sim + 4\, p\sim^{3}\, CWmin\sim \lambda\sim + 2\, p\sim^{3}\, DIFS\sim \lambda\sim + 2\, p\sim^{7}\, DIFS\sim \lambda\sim$$

$$+ 2\, \lambda\sim SIFS\sim + 2\, l4\sim + 2\, \lambda\sim DIFS\sim + 2\, \lambda\sim Ed_ACK\sim + 2\, p\sim^{5}\, l4\sim + 2\, p\sim^{3}\, l4\sim + 2\, p\sim^{4}\, l4\sim + 2\, p\sim^{6}\, l4\sim + 2\, p\sim^{7}\, l4\sim + 2\, p\sim l4\sim$$

$$\left.+ 2\, p\sim^{2}\, l4\sim\right)\Big)$$

```
> #temps d'authentification, sa moyenne :
  phi:=simplify(n*(muROTd_1+muROTd_2));
```

$$\phi := \frac{1}{\lambda\sim}\Big(n\sim hops\sim\left(4\, p\sim^{3}\, ACK_Timeout\sim \lambda\sim + 4\, p\sim ACK_Timeout\sim \lambda\sim + 2\, p\sim CWmin\sim \lambda\sim + 4\, p\sim DIFS\sim \lambda\sim + 4\, p\sim^{5}\, ACK_Timeout\sim \lambda\sim \right. \quad \textbf{(10}$$

$$+ 4\, p\sim^{7}\, ACK_Timeout\sim \lambda\sim + 128\, p\sim^{7}\, CWmin\sim \lambda\sim + 4\, p\sim^{4}\, ACK_Timeout\sim \lambda\sim + 16\, p\sim^{4}\, CWmin\sim \lambda\sim + 4\, p\sim^{4}\, DIFS\sim \lambda\sim$$

$$+ 32\, p\sim^{5}\, CWmin\sim \lambda\sim + 4\, p\sim^{5}\, DIFS\sim \lambda\sim + 4\, p\sim^{2}\, ACK_Timeout\sim \lambda\sim + 4\, p\sim^{2}\, CWmin\sim \lambda\sim + 4\, p\sim^{2}\, DIFS\sim \lambda\sim + 4\, p\sim^{6}\, DIFS\sim \lambda\sim$$

$$+ 4\, p\sim^{6}\, ACK_Timeout\sim \lambda\sim + 64\, p\sim^{6}\, CWmin\sim \lambda\sim + 8\, p\sim^{3}\, CWmin\sim \lambda\sim + 4\, p\sim^{3}\, DIFS\sim \lambda\sim + 4\, p\sim^{7}\, DIFS\sim \lambda\sim + 4\, \lambda\sim SIFS\sim + l4\sim$$

$$+ l2\sim + l1\sim + l3\sim + 4\, \lambda\sim DIFS\sim + 4\, \lambda\sim Ed_ACK\sim + p\sim^{3}\, l3\sim + p\sim^{6}\, l3\sim + p\sim^{4}\, l3\sim + p\sim^{5}\, l4\sim + p\sim^{3}\, l4\sim + p\sim^{4}\, l4\sim + p\sim^{6}\, l4\sim$$

$$+ p\sim^{7}\, l4\sim + p\sim^{4}\, l2\sim + p\sim l2\sim + p\sim^{2}\, l3\sim + p\sim^{5}\, l3\sim + p\sim^{7}\, l3\sim + p\sim^{6}\, l2\sim + p\sim l4\sim + p\sim l3\sim + p\sim^{2}\, l4\sim + p\sim^{5}\, l2\sim + p\sim^{7}\, l2\sim + p\sim^{3}\, l2\sim$$

$$+ p\sim^{2}\, l1\sim + p\sim^{5}\, l1\sim + p\sim^{7}\, l1\sim + p\sim^{3}\, l1\sim + p\sim^{4}\, l1\sim + p\sim^{6}\, l1\sim + p\sim l1\sim + p\sim^{2}\, l2\sim)\,)$$

```
> Valeurs_1:=[DIFS=10.0E-06+2*20.0E-06, CWmin=32*20.0E-06, SIFS=10.0E-06, ACK_Timeout=334.0E-06,
  Ed_ACK=304.0E-06, lambda=(11.0E06/8), hops=3, l1=287, l2=32, l3=1593, l4=1925, p=0.1]: # DIFS,
  SIFS, ACK_Timeout, Ed_ACK, and CWmin=random(1..cwmin)*slotTime are in secondes. lambda is in bytes
  per s, l in bytes.
> Valeurs_2:=[DIFS=10.0E-06+2*20.0E-06, CWmin=32*20.0E-06, SIFS=10.0E-06, ACK_Timeout=334.0E-06,
  Ed_ACK=304.0E-06, n=5, hops=3, l1=287, l2=32, l3=1593, l4=1925, p=0.1]:
> Valeurs_3:=[DIFS=10.0E-06+2*20.0E-06, CWmin=32*20.0E-06, SIFS=10.0E-06, ACK_Timeout=334.0E-06,
  Ed_ACK=304.0E-06, n=5, lambda=(11.0E06/8), l1=287, l2=32, l3=1593, l4=1925, p=0.1]:
> Valeurs_5:=[DIFS=10.0E-06+2*20.0E-06, CWmin=32*20.0E-06, SIFS=10.0E-06, ACK_Timeout=334.0E-06,
  Ed_ACK=304.0E-06, n=5, lambda=(11.0E06/8), hops=3, l1=287, l2=32, l3=1593, l4=1925]:
> f_1:=eval(simplify(subs(Valeurs_1, phi))):
> f_2:=eval(simplify(subs(Valeurs_2, phi))):
> f_3:=eval(simplify(subs(Valeurs_3, phi))):
> f_5:=eval(simplify(subs(Valeurs_5, phi))):
> L_1:=[seq([i,subs(n=i,f_1)],i=1..6)]:
> #courbe du temps moyen d'authentification (secondes) en fonction de n :
  plot(L_1);
```

# Bibliographie


[1] C. M. Chlamtac et J. J.-M. Liu : Mobile ad hoc networking : imperatives and challenges. *Ad Hoc Networks*, pages 13–64, jul 2003.

[2] Organisation de coopération et de développement économique (ocde). http://www.oecd.org/.

[3] ATILF : Le Trésor de la Langue Française informatisé. http://atilf.atilf.fr/tlf.htm.

[4] MANET-WG : IETF Mobile Ad-hoc Networks WG. http://datatracker.ietf.org/wg/manet/.

[5] S. Corson et J. Macker : Mobile Ad hoc Networking (MANET) : Routing Protocol Performance Issues and Evaluation Considerations (RFC 2501). *IETF working group on Mobile Adhoc Networks*, Jan 1999.

[6] C. Bidan : Contributions à la sécurité des réseaux auto-organisés (HDR Dissertation), jun 2010.

[7] NIST : Wireless Ad Hoc Network Projects. http://w3.antd.nist.gov/wahn_home.shtml, jun 2008.

[8] MOTOROLA : Mesh wide area networks. http://www.meshnetworks.com.

[9] SPAN-works : Spanworks home. http://www.spanworks.com.

[10] T. Clausen et P. Jacquet : Optimized Link State Routing Protocol (OLSR) (RFC 3626). http://tools.ietf.org/html/rfc3626, oct 2003.

[11] R. Ogier, F. Templin et M. Lewis : Topology Dissemination Based on Reverse-Path Forwarding (TBRPF) (RFC 3684). http://tools.ietf.org/html/rfc3684, feb 2004.

[12] C. Perkins, E. Belding-Royer et S. Das : Ad hoc On-Demand Distance Vector (AODV) Routing (RFC 3561). http://tools.ietf.org/html/rfc3561, jul 2003.

[13] D. Johnson, Y. Hu et D. Maltz : The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4 (RFC 4728). http://tools.ietf.org/html/rfc4728, feb 2007.

[14] I. Chakeres et C. Perkins : Dynamic MANET On-demand (DYMO) Routing (draft-ietf-manet-dymo-21). http://tools.ietf.org/html/draft-ietf-manet-dymo-21, jul 2010.

[15] T. Ramrekha, E. Panaousis et C. Politis : ChaMeLeon (CML) : A hybrid and adaptive routing protocol for Emergency Situations (draft-ramrekha-manet-cml-01.txt). http://tools.ietf.org/id/draft-ramrekha-manet-cml-01.txt, aug 2010.

[16] Macker, J. and SMF Design Team in the IETF MANET WG : Simplified Multicast Forwarding (draft-ietf-manet-smf-10). http://tools.ietf.org/html/draft-ietf-manet-smf-10, sep 2010.

[17] DARPA : DARPA Home Page. http://www.darpa.mil/.

[18] J. Jubin et J.D. Tornow : The DARPA Packet Radio Network Protocols. *In Proceeding of the IEEE*, volume 75, Jan 1987.

[19] W.C. Fifer et F.J. Bruno : The Low-Cost Packet Radio. *Proceedings of the IEEE*, 75(1):33–42, January 1987.

[20] B. Leiner, R. Ruth et A.R. Sastry : Goals and challenges of the DARPA GloMo program. *IEEE Personal Communications*, 3(6):34–43, 1996.

[21] University XXI : Force XXI Battle Command Brigade And Below (FBCB2), January 2001.

[22] Information Theory for Mobile Ad-Hoc Networks (ITMANET). http://www.darpa.mil/i2o/programs/itmanet/itmanet.asp, 2008.

[23] J. LAFFAY : Internet Tactique. *RAIDS Magazine*, jul 2002.

[24] L. Canourgues : *Algorithmes de Routage dans les Réseaux Mobile Ad hoc Tactique à Grange Echelle.* Thèse de doctorat, Université de Toulouse III - Paul Sabatier, may 2008.

[25] THALES : Mobile Tactical Internet - M@tis. http://www.thalesgroup.com/Pages/Solution.aspx?id=7540&pid=1195.

[26] IEEE Computer Society : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Standard 802.11, jun 1999.

[27] J. Kruys : Hiperlan, applications and requirements. *In Third IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 1992. Proceedings, PIMRC'92.*, pages 133–138, 1992.

[28] NIST : Mobile Ad hoc Data Networks for Emergency Preparedness Telecommunications. http://w3.antd.nist.gov/wctg/manet/manet.html#NCS, feb 2000.

[29] P. Pouschat : Projet Esponder. http://www.valabre-ceren.org/fr/projet-detail.php?projet_type=2\&projet_id=7.

[30] Boulapoire : PlayStation, les chiffres de vente. http://www.gamekult.com/actu/gc-playstation-les-chiffres-de-vente-A0000078776.html.

[31] ITU-R : Requirements related to technical performance for IMT-Advanced radio interface(s). Rapport technique M.2134, nov 2008.

[32] S. Frattasi, H. Fathi, F.H.P Fitzek, R. Prasad et M.D. Katz : Defining 4G Technology from the User's Perspective. *IEEE Network*, Jan/feb 2006.

[33] K. Marinova : Milen Nikolov Sends the Mobile Phone in the Outer Space. http://paper.standartnews.com/en/article.php?d=2009-01-06\&article=26293, Jan 2009.

[34] B. McCarthy, C. Edwards et M. Dunmore : Using NEMO to support the global reachability of MANET nodes. *In IEEE INFOCOM*, pages 2097–2105. IEEE, 2009.

[35] EURESCOM : Integration of mobile ad-hoc networks, EU project DAIDALOS. `http://ftp.eurescom.de/message/messageMar2005/Integration_of_mobile_ad_hoc_netwroks_%20EU_Project_DAIDALOS.asp`, 2008.

[36] l. Linda Kaällström : Seamless Service Interworking in Heterogeneous Mobile and Ad-Hoc Networks (SESSI). `www.tml.tkk.fi/Research/SESSI`, 2005.

[37] Dong-Won Kum, Jin-Su Park, You-Ze Cho et Byoung-Yoon Cheon : Performance evaluation of AODV and DYMO routing protocols in MANET. *In CCNC'10 : Proceedings of the 7th IEEE conference on Consumer communications and networking conference*, pages 1046–1047, Piscataway, NJ, USA, 2010. IEEE Press.

[38] J.P. Hubaux, L. Buttyán et S. Capkun : The Quest for Security in Mobile Ad Hoc Networks. *In Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*. ACM, 2001.

[39] P. Michiardi : Coopération dans les réseaux ad-hoc : Application de la théorie des jeux et de l'évolution dans le cadre d'observabilité imparfaite. *In Symposium sur la Sécurité des Technologies de l'Information et des Communications 2006*, pages 286–299. École Supérieure et d'Application des Transmissions, 2006.

[40] E. Littré : Dictionnaire de français Littré. `http://littre.reverso.net/dictionnaire-francais`, 1863-1877.

[41] P. Larousse *et al.* : Larousse dictionnaire de français. `http://www.larousse.fr/dictionnaires/francais`, 1854.

[42] P. Robert, A. Rey *et al.* : Le Grand Robert de la langue française.

[43] D.L. Parnas : On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058, 1972.

[44] A.J. Menezes, P.C. Van Oorschot et S.A. Vanstone : *Handbook of applied Cryptography*, chapitre 1, page 3. Crc Press, 1997.

[45] ANSSI : Référentiel Général de Sécurité (RGS). JORF du 18 mai 2010, may 2010.

[46] A. Tanenbaum : *Réseaux*, volume 4. Pearson, Education France, 2004.

[47] A.J. Menezes, P.C. Van Oorschot et S.A. Vanstone : *Handbook of applied Cryptography*. Crc Press, 1997.

[48] BlueKrypt : Keylength - Compare all Methods. `http://www.keylength.com/en/compare/`, 2011.

[49] A. Shamir : How to Share a Secret. *In Communications of the ACM*, volume 22, pages 612–613, nov 1979.

[50] AAA-WG : IETF Authentication, Authorization and Accounting Working Group. `http://datatracker.ietf.org/wg/aaa/charter`, 2011.

[51] B. Aboba : The IEEE 802 / IETF Relationship (Internet-draft). `http://www.iab.org/documents/drafts/draft-iab-ieee-802-rel-01.txt`, aug 2005.

[52] IRTF : Authentication Authorisation Accounting Architecture Research Group (AAAARCH). `http://irtf.org/concluded/aaaarch`, 2011.

[53] C. Perkins : IP Mobility Support for IPv4 (RFC 3344). http://tools.ietf.org/html/rfc3344, aug 2002.

[54] D. Johnson, C. Perkins et J. Arkko : Mobility Support in IPv6. http://tools.ietf.org/html/rfc3775, jun 2004.

[55] C. Rigney, S. Willens, A. Rubens et W. Simpson : Remote Authentication Dial In User Service (RADIUS) (RFC 2865). *IETF*, jun 2000.

[56] P. Calhoun, J. Loughney, E. Guttman, G. Zorn et J. Arkko : Diameter Base Protocol (RFC 3588). *IETF*, sep 2003.

[57] C. Neuman, T. Yu, S. Hartman et K. Raeburn : The Kerberos Network Authentication Service (V5) (RFC 4120). http://www.ietf.org/rfc/rfc4120.txt, jul 2005.

[58] C. Rigney : RADIUS Accounting. http://tools.ietf.org/html/rfc2866, jun 2000.

[59] Finseth C. : An Access Control Protocol, Sometimes Called TACACS (RFC 1492). http://tools.ietf.org/html/rfc1492, jul 1993.

[60] Nerim : Nerim - Accueil. http://www.nerim.fr.

[61] S. Sargento, T. Calçada, J.P. Barraca, S. Crisóstomo, J. Girão, M. Natkaniec, N. Vicari, F.J. Galera, M. Ricardo et A. Glowacz : Mobile ad-hoc networks integration in the DAIDALOS architecture. *In Proceedings of the 14th IST Mobile & Wireless Communications Summit*, Germany, jun 2005.

[62] H. Chaouchi et M. Laurent-Maknavicius : SAACCESS : Secured Ad hoc ACCess framework. *In International Conference on New Technologies, Mobility and Security NTMS'07*, pages 425–436. Springer, 2007.

[63] G. Kounga et C. Schaefer : Selling multimedia resources in ad hoc networks. *In IEEE Communications Magazine*, volume 46, pages 126 – 131, feb 2008.

[64] H. Moustafa, J. Forestier et M. Chaari : Distributed authentication for services commercialization in ad hoc networks. *In Mobility '09 : Proceedings of the 6th International Conference on Mobile Technology, Application & Systems*, pages 1–8, New York, NY, USA, 2009. ACM.

[65] S. Leggio, S. Liimatainen, J. Manner, T. Mikkonen, J. Saarinen et A. Ylä-Jääski : Towards service interworking among ad-hoc networks and the internet. *the 14th IST Mobile and Wireless Communications Summit, Dresden*, 2005.

[66] S. Zhong, J. Chen et Y.R. Yang : Sprite : A simple, Cheat-Proof, Credit-Based System for Mobile Ad-Hoc Networks. *In INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1987–1997. IEEE, 2003.

[67] P. Lamsal : AAA and PKI in Ad Hoc Networks.

[68] P. Lamsal : Distributed AAA : Proposals for ad hoc networks, 2004.

[69] A.R. Khakpour, M. Laurent-Maknavicius et H. Chaouchi : WATCHMAN : An Overlay Distributed AAA Architecture for Mobile Ad hoc Networks. *In Proc. IEEE Third International Conference on Availability, Reliability and Security (ARES 2008)*, pages 144–152, Barcelona, Spain, mar 2008.

[70] J. Arkko, J. Kempf, B. Zill et P. Nikander : SEcure Neighbor Discovery (SEND) (rfc 3971). `http://tools.ietf.org/html/rfc3971`, mar 2005.

[71] E. Baccelli et M. Townsley : Ip addressing model in ad hoc networks. RFC 5889, sep 2010.

[72] L. Buttyán et J.P. Hubaux : Enforcing service availability in mobile ad-hoc WANs. *In IEEE/ACM First Annual Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC)*, pages 87–96, Boston, MA, USA, aug 2000.

[73] Autoconf-WG : IETF Ad-Hoc Network Autoconfiguration WG. `http://datatracker.ietf.org/wg/autoconf/`.

[74] T. Aura : Cryptographically Generated Addresses (CGA) (RFC 3972). *IETF*, mar 2005.

[75] Moustafa, H. and Bourdon, G. and Gourhant, Y. : Authentication, authorization and accounting (AAA) in hybrid ad hoc hotspot's environments. *In Proceedings of the 4th international workshop on Wireless mobile applications and services on WLAN hotspots*, WMASH '06, pages 37–46, New York, NY, USA, sep 2006. ACM.

[76] R. Ostrovsky et M. Yung : How to withstand mobile virus attacks (extended abstract). *In Proceedings of the tenth annual ACM symposium on Principles of distributed computing*, pages 51–59. ACM, 1991.

[77] Mohammed, N. and Otrok, H. and Wang, Lingyu and Debbabi, M. and Bhattacharya, P. : Mechanism Design-Based Secure Leader Election Model for Intrusion Detection in MANET. 8(1):89–103, 2011.

[78] Ghamri-Doudane, Y.M. and Senouci, S.M. and Agoulmine, N. : P-SEAN : A Framework for Policy-based Server Election in Ad hoc Networks. pages 271–281. IEEE, 2006.

[79] Telecommunication Standardization Sector of ITU : Information technology – Open Systems Interconnection – The Directory : Public-key and attribute certificate frameworks. *In ITU-T Recommendation X.509*, SERIES X : DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY. ITU, aug 2005.

[80] ISO : ISO rechercher [9798-3]. `http://www.iso.org/iso/fr/search.htm?qt=9798-3&published=on&active_tab=standards`.

[81] V. Shoup : Practical Threshold Signatures. *In EUROCRYPT 2000*, volume 1807, pages 207–220, 2000.

[82] S. Larafa, M. Laurent-Maknavicius et H. Chaouchi : Light and Distributed AAA Scheme for Mobile Ad-hoc Networks. *In Proc. First Workshop on Security of Autonomous and Spontaneous Networks (SETOP 2008)*, pages 93–103, Loctudy, France, oct 2008.

[83] S. Larafa et M. Laurent-Maknavicius : Protocols for Distributed AAA Framework in Mobile Ad-hoc Networks. *In Proc. Workshop on Mobile and Wireless Networks Security (MWNS 2009)*, pages 75–86, Aachen, Germany, may 2009.

[84] S. Larafa et M. Laurent : Towards Multiple-Exchange Protocol Use in Distributed AAA Frameworks for More Autonomy in MANETs. *In In-*

*ternational Symposium on Computers and Communications, ISCC 2011*, Corfou, Grèce, jun 2011.

[85] W. Diffie et M.E. Hellman : New directions in cryptography. *In Proc. IEEE Transactions on Information Theory*, pages 644–654, nov 1976.

[86] Cousot, P. and others : The Astrée Static Analyzer. `http://www.astree.ens.fr`, 2011.

[87] R. Jacquart, éditeur. *Building the Information Society*. Kluwer Academic Publishers, 2004.

[88] R. Lai et A. Jirachiefpattana : *Communication Protocol Specification and Verification*. Springer, 1998.

[89] IST : ISTweb-IST Projects. `http://cordis.europa.eu/fetch?CALLER=PROJ_IST&ACTION=D&RCN=66394&DOC=1&CAT=PROJ&QUERY=1`, 2007.

[90] IST : the AVISPA Project. `http://www.avispa-project.org/`, 2007.

[91] A. Gupta et S. Malik, éditeurs. *Computer Aided Verification*. Lecture Notes in Computer Science. 2008.

[92] The-AVISPA-team : *HLPSL Tutorial*, jun 2006.

[93] D. Dolev et A. Yao : On the Security of Public-Key Protocols. *IEEE Transactions on Information Theory*, 1983.

[94] Pouchain M. et M. Covacho : Validation formelle d'un protocole d'authentification distribuée dans les réseaux ad-hoc. Rapport technique, Télécom SudParis, jun 2010.

[95] An, G. and Kim, K. and Jang, J. and Jeon, Y. : Analysis of SEND Protocol through Implementation and Simulation. *In Proceedings of the 2007 International Conference on Convergence Information Technology*, ICCIT '07, pages 670–676, Washington, DC, USA, 2007. IEEE Computer Society.

[96] R. Zhang, Y. Zhang et X. Huang : JR-SND : Jamming-resilient secure neighbor discovery in mobile ad hoc networks. *In Proceedings of the International Conference on Distributed Computing Systems*, ICDCS 2011, Minneapolis, Minnesota, jun 2011.

[97] P. Papadimitratos, M. Poturalski, P. Schaller, P. Lafourcade, D. Basin, S. Čapkun et J.-P. Hubaux : Secure Neighborhood Discovery : A Fundamental Element for Mobile Ad Hoc Networking. *IEEE Communications Magazine*, 46(2):132–139, Février 2008.

[98] L. II Hanzo et R. Tafazolli : Admission Control Schemes for 802.11-Based Multi-Hop Mobile Ad hoc Networks : A Survey. *IEEE Communications Surveys & Tutorials*, 11(4):78–108, 2009.

[99] D. Zwillinger et S. Kokoska : *CRC Standard Probability and Statistics Tables and Formulae*. CRC Press, 2000.

[100] M. Kendall, A. Stuart et J.K. Ord : *The Advanced Theory of Statistics*, volume 1. Charles Griffin & Company Limited, 1977.

[101] M. Abramowitz et I.A. Stegun : *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, New York, 1964.

[102] M. Kendall, A. Stuart et J.K. Ord : *The Advanced Theory of Statistics*, volume 1, chapitre 14, page 348 (formule 14.4). Charles Griffin & Company Limited, 1977.

[103] A. Sotirov, M. Stevens, J. Appelbaum, A. Lenstra, D. Molnar, D.A. Osvik et B. de Weger : Creating a rogue CA certificate. `http://www.win.tue.nl/hashclash/rogue-ca/`.

[104] UC Berkeley, LBL, USC/ISI et Xerox PARC : The NS Manual (formerly ns Notes and Documentation). `http://www.isi.edu/nsnam/ns/ns-documentation.html`, Jan 2009.

[105] Sourceforge : OTCL Home Page. `http://otcl-tclcl.sourceforge.net/otcl/`.

[106] Python Software Foundation : Python Programming Language – Official Website. `http://www.python.org/`, 2011.

[107] S. Larafa et M. Laurent : Authentication protocol runtime evaluation in distributed AAA framework for Mobile Ad-Hoc Networks. *In 2010 IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS)*, pages 277–281, jun 2010.

[108] G. Fleury, P. Lacomme et A. Tanguy : *Simulation à événements discrets*. Eyrolles, nov 2006.

[109] D.A. Takus et D.M. Profozich : Arena software tutorial. *In Proceedings of the 1997 Winter Simulation Conference*. ed. S. Andradottir, K.J. Healy, D.H. Withers, and B.L. Nelson, 1997.

[110] O.-J. Dahl et K. Nygaard : Simula : an algol-based simulation language. *Commun. ACM*, 9(9):671–678, 1966.

[111] C.D. Pegden : Introduction to SIMAN. *In WSC '83 : Proceedings of the 15th conference on Winter simulation*, pages 231–241, Piscataway, NJ, USA, 1983. IEEE Press.

[112] A. Jean-Marie : Le langage QNAP2 pour la simulation à évènements discrets. Projet Mistral, INRIA Sophia-Antipolis, Décembre 1998.

[113] NS-3 Group : NS-3, an INRIA, NSF, UW and GT Project. `http://www.nsnam.org`.

[114] OPNETTechnologies Inc. : Network Modeling | Network Simulation. `http://www.opnet.com/solutions/network_rd/modeler.html`.

[115] OMNeT++ Community : OMNeT++ Network Simulation Framework. `http://www.omnetpp.org`, 2009.

[116] UCLA-PCL : About GloMoSim. `http://pcl.cs.ucla.edu/projects/glomosim`.

[117] QualNet : QualNet Developer | network medeling & simulation software | Scalable Network Technologies. `http://www.scalable-networks.com/products/qualnet/`, 2011.

[118] NSL : SIMREAL technology Homepage. `http://nsl10.cs.nctu.edu.tw`, 2002.

[119] S. Kurkowski, T. Camp et M. Colagrosso : Manet simulation studies : the incredibles. *SIGMOBILE Mob. Comput. Commun. Rev.*, 9(4):50–61, 2005.

[120] E. Weingärtner, H. Vom Lehn et K. Wehrle : A performance comparison of recent network simulators. *In ICC'09 : Proceedings of the 2009 IEEE international conference on Communications*, pages 1287–1291, Piscataway, NJ, USA, 2009. IEEE Press.

[121] S. Keshav : REAL 5.0 Overview. http://www.cs.cornell.edu/skeshav/real/overview.html, aug 1997.

[122] A. Helmy et S. Kumar : VINT : Virtual InterNetwork Testbed . http://www.isi.edu/nsnam/vint/index.html, oct 1997.

[123] Berkeley-Lab : Lawrence Berkeley National Laboratory. http://www.lbl.gov, may 2011.

[124] UC-Regents : University of Califormia, Berkeley. http://berkeley.edu, 2011.

[125] USC : Information Sciences Institute - Welcome. http://www3.isi.edu/home, 2011.

[126] K. Lan : SAMAN : Simulation Augmented by Measurement and Analysis for Networks. http://www.isi.edu/saman/index.html, jul 2001.

[127] X. Chen : Collaborative Simulation for Education and Research CONSER. http://www.isi.edu/conser/index.html, mar 2002.

[128] The ICSI Networking Group. http://www.icir.org/, feb 2011.

[129] R. Katz et E. Brewer : The UC Berkeley BARWAN Research Project CDROM. http://daedalus.cs.berkeley.edu.

[130] The Rice University Monarch Project. http://www.monarch.cs.rice.edu/, oct 2004.

[131] CMU : Carnegie Mellon University. http://www.cmu.edu/index.shtml, 2011.

[132] P. L'Ecuyer : Good parameters and implementations for combined multiple recursive random number generators. *Operations Research*, 47(1): 159–164, 1999.

[133] NS-2 Class Hierarchy. http://www.isi.edu/nsnam/nsdoc-classes/hierarchy.html, 2004.

[134] Red Hat, Inc. : Cygwin. http://www.cygwin.com, 2011.

[135] Riley, G. and Park, A. : PDNS - Parallel/Distributed NS. http://www.cc.gatech.edu/computing/compass/pdns, 2004.

[136] E. Chen et C. Ng : Comparative Analysis of Wireless Routing Algorithms in ns-2. 2006.

[137] S. Ivanov, A. Herms et G. Lukas : Experimental validation of the ns-2 wireless model using simulation, emulation, and real network. *In Proceedings of the 4th Workshop on Mobile Ad-Hoc Networks (WMAN 2007)*. Citeseer, 2007.

[138] H.P. McKean : *Stochastics Integral*. Academic Press, 1969.

[139] J.L. Boob : *Stochastic Processes*, chapitre 10, pages 464–473. John Willey & Sons, Inc., 1953.

[140] C. Bettstetter : Smooth is better than sharp : a random mobility model for simulation of wireless networks. *In MSWIM'01 : Proceedings of the 4th ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, pages 19–27, New York, NY, USA, 2001. ACM.

[141] J. Yoon, M. Liu et B. Noble : Random waypoint considered harmful. *In IEEE INFOCOM*, volume 2, pages 1312–1321. Citeseer, 2003.

[142] W. Navidi, T. Camp et N. Bauer : Improving the accuracy of random waypoint simulations through steady-state initialization. *In 15th International Conference on Modeling and Simulation (MS'04)*, pages 319–326, mar 2004.

[143] W. Xiuchao : Simulate 802.11 b channel within ns2. *National University of Singapore, Singapore*, 2004.

[144] J. Chung et M. Claypool : *NS by Example.* Worcester Polytechnic Institute, 2006.

[145] J. Robinson : Making ns-2 simulate an 802.11b link. apr 2005.

[146] M. Becker et A.L. Beylot : *Simulation des réseaux*, chapitre 3, pages 75–79. LAVOISIER, 2006.

[147] P. Gaudry et É. Schost : Le cryptosystème hyperelliptique Surf1271, jul 2009.

[148] Perkins, C.E. and Royer, E.M. : Ad-Hoc On-Demand Distance Vector Routing. *In Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications*, WMCSA '99, pages 90 – 100, Washington, DC, USA, 1999. IEEE Computer Society.

[149] J. Lelong-Ferrand et Arnaudiès J.-M. : *Cours de mathématiques*, volume 1 : Algèbre. Dunod, mar 2003.

[150] A. Blanchard : *Les corps non commutatifs.* PUF, 1972.

[151] P.-A. Fouque : *Le partage de clés cryptographiques : Théorie et Pratique.* Thèse de doctorat, Université Paris 7, 2001.

[152] P. Naudin et C. Quitté : *Algorithmique algébrique.* Masson, 1992.

[153] D. Eastlake et P. Jones : US Secure Hash Algorithm 1 (SHA1) (RFC 3174). `http://www.ietf.org/rfc/rfc3174.txt`, sep 2001.

[154] I. Déchène : The Picard Group, or how to build a group from a set. *In Tutorial on Elliptic and Hyperelliptic Curve Cryptography*, University College Dublin, Ireland, sep 2007.

[155] A. Lenstra et E. Verheul : The XTR public key system. *In Advances in Cryptology—CRYPTO 2000*, pages 1–19. Springer, 2000.

[156] S. Deering et R. Hinden : Internet Protocol, Version 6 (IPv6) Specification (RFC 2460). `http://tools.ietf.org/html/rfc2460`, dec 1998.

[157] S. Thomson et T. Narten : IPv6 Stateless Address Autoconfiguration (RFC 2462). `http://tools.ietf.org/html/rfc2462`, dec 1998.

[158] C.E. Perkins, J.T. Malinen, R. Wakikawa, Elizabeth M. Belding-Royer et Y. Sun : IP Address Autoconfiguration for Ad Hoc Networks (INTERNET DRAFT). `http://www.cs.ucsb.edu/~ebelding/txt/autoconf.txt`, nov 2001.

[159] J. Lee, S. Ahn, H. Yu, Y.S. Kim et J.S. Jin : Adress Autoconfiguration and Route Determination Mechanisms for the MANET Architecture Overcoming the Multi-link Subnet Model. *In Proceedings of the 23rd international conference on Information Networking*, ICOIN'09, pages 122–126, Piscataway, NJ, USA, 2009. IEEE Press.

[160] T. Narten, E. Nordmark et W. Simpson : Neighbor Discovery for IP Version 6 (IPv6) (RFC 2461). `http://tools.ietf.org/html/rfc2461`, dec 1998.

[161] M. Grajzer : ND++ - an extended IPv6 Neighbor Discovery protocol for enhanced duplicate address detection to support stateless address autoconfiguration in IPv6 mobile ad hoc networks (Internet Draft). `http://tools.ietf.org/html/draft-grajzer-autoconf-ndpp-00`, mar 2011.

[162] P. Nikander, J. Kempf et E. Nordmark : IPv6 Neighbor Discovery (ND) Trust Models and Threats (RFC 3756). `http://tools.ietf.org/html/rfc3756`, may 2004.

[163] B. Kaliski : PKCS #1 : RSA Encryption Version 1.5 (RFC 2313). `http://tools.ietf.org/html/rfc2313`, mar 1998.

[164] G. Bianchi : Performance analysis of the IEEE 802. 11 Distributed Coordination Function. *IEEE Journal on Selected Areas in Communications*, 18(3):535–547, mar 2000.

# Index


Établissement de clés de session, 53
élection, 59
*Clé de session*, 201

AA, 28
AAA, 45
AAA Authentication AVP, 95
accounting, 48
attaque, 26
Attribute Value Pair, AVP, 95
Authentification, 52
authentification, 48
Authentification et Autorisation, 28
Autorisation, 54
autorisation, 48

bootstrapping, 52

canal sécurisé, 41
certificat, 42
chaîne de confiance, 41
chiffrement, 35
Clé
    de chiffrement, 35
    de déchiffrement, 35
clé de session, 53
codage, 35
cryptage, 35
Cryptographically Generated Address, CGA, 207
cryptographie à seuil, 42

Diffie-Hellman, 42
droits d'accès, 30
Duplicate Address Detection, DAD, 204

EAP (*Extensible Authentication Protocol*), 53
empreinte, 35
enrollment, 52
entités, 26

hachage, 35
hashing, 35
hiérarchisation des relations de confiance, 41

IKE (*Internet Key Exchange*), 42
IKE (Internet Key Exchange), 45
initialisation, 52
inscription, 52
IPv6, 203

mécanisme d'authentification, 52

Neighbor Discovery Protocol, NDP, 205
non répudiation, 48

partage de secrets, 41
PGP (*pretty good privacy*), 42
protocole, 25

question-réponse, 39

réseau, 23
Réseau ad-hoc, xix, 4
relation de confiance, 41
RGS (Référentiel Général de Sécurité), 22

Sécurité, 22
sécurité, 23
SEcure Neighbor Discovery, SEND, 203
service de sécurité, 44
Session autorisée, 54
seuil cryptographique, 43

terminaux ad hoc, 4
traçabilité, 54
traçage, 54