Projet N° ANR-05-RNRT-028-01

SEREADMO

Sécurité des Réseaux Ad hoc & Mojette

Type : Exploratoire

Thématique ANR : RNRT 2005- thème « sécurité » « mobilité et interopérabilité » « intelligence ambiante »

D 61 – Description de la plate-forme d'expérimentation

Date de Livraison du délivrable : xx/xx/2007
Date de soumission du délivrable : xx/xx/2008

Démarrage du Projet : 07/2006 Durée : 36 mois

Pilotage et Gestion du projet SEREADMO (Main leader) :
THALES Communications – France

Révision : Release v.0.1

Niveau de confidentialité		
PU	Public	X
PP	Diffusion Restreinte aux autres participants ANR RNRT 2005 (incluant la commission RNRT)	
RE	Diffusion Restreinte aux groupes spécifiés par le consortium (incluant la commission RNRT)	
CO	Confidentiel aux seuls membres du consortium (incluant la commission RNRT)	

Résumé

Ce document permet de spécifier la plate-forme de validation et les expérimentations réalisées dans le cadre du projet SEREADMO.

Responsabilités

	Sociétés	Nom	Mail
Approbation Main Leader	THALES	GRALL Eric	eric.grall@fr.thalesgroup.com
Approbation Consortium (Majorité)	KEOSYS, IRCCyN, SIC		

Evolutions successives

Indice	Date	Modifié par	Nature de l'évolution
Draft v.0.1	07/05/2008	P.Lesage	Création Version Préliminaire

TABLE DES MATIERES

TABLE DES FIGURES

1. ACRONYMES ET DEFINITIONS

Abréviation	Définition
API	Application Programming Interface
IP	Internet Protocol
MPOLSR	Multi-Path OLSR
NDIS	Network Driver Interface Specification
OLSR	Optimized Link State Routing Protocol
OSI	Open Systems Interconnection
PCMCIA	Personal Computer Memory Card International Association
SEREADMO	Sécurité des Réseaux Ad hoc & Mojette
SERANO	SEcure Redundancy Adhoc NetwOrk
TCP	Transmission Control Protocol
USB	Universal Serial BUS
UDP	User Datagram Protocol

2. PROCEDURES

2.1 Documents internes référencés

Les documents internes au projet SEREADMO cités dans ce document sont les suivants :

D12 – Spécification technique de l'architecture du protocole

D14 – Résultats de simulation : routage Mojette
D15 – Résultats de simulation : QoS

D21 – Etat de l'art « Sécurité des réseaux Ad Hoc »
Part 1 – Vision globale des technologies et Recherches actuelles sur la sécurité des Réseaux ad hoc
Part 2 – Cible de Sécurité SEREADMO
D22 – Spécification du protocole de sécurité SEREADMO : SERANO

2.2 Confidentialité

Ce document possède un niveau de confidentialité « PUBLIC ».

3. DESCRIPTION DES RESULTATS

Ce document permet de définir la plate-forme d'expérimentation mise en œuvre pour la validation terrain du protocole SEREADMO, sur les bases des spécifications réalisées dans les délivrables D12 (Spécification technique de l'architecture du protocole) et D22 (Spécification du protocole de sécurité SEREADMO : SERANO).

Ce document permet également de définir les différentes expérimentations qui seront réalisées pour valider ce protocole et confronter les retours terrains aux simulations réalisées sous NS2, décrites dans les délivrables D14 (Résultats de simulation : routage Mojette) et D15 (Résultats de simulation : QoS)

4. VISION ET CONTEXTE DU DELIVRABLE

4.1 Description du projet SEREADMO

De nos jours, les communications téléphoniques sans fils se sont imposées sur le marché et en parallèle, les réseaux locaux radio prennent de plus en plus d'importance. L'aire Post-PC a ainsi vu le développement de petits ordinateurs portables avec une grande variété de services : les exemples les plus courants sont bien sur les téléphones cellulaires (Smart phone), et les PDAs (Assistant Personnel). Le nomadisme entraîne de nouvelles habitudes et de nouveaux besoins en matière de services, et s'accompagne de nouveaux concepts réseaux, dans lesquels s'inscrivent les réseaux Ad Hoc et les réseaux Mesh.

Un réseau Mesh est un réseau maillé ne disposant pas d'une infrastructure fixe et dans lequel tous les nœuds participent au routage. Un réseau Ad Hoc est un réseau Mesh « spontané » et sans fil, pour lequel les hôtes mobiles doivent former d'une manière ad hoc, au gré de leur apparition dans le réseau, une architecture de réseau globale qui peut être utilisée comme infrastructure du système.

Les réseaux sans fil ont à prendre en compte des exigences de qualité de service et de sécurité plus difficile à mettre en oeuvre que pour des réseaux filaires. La gestion de l'acheminement de données (ou routage) dans un réseau Ad Hoc, consiste à garantir, à n'importe quel moment, la connexion entre n'importe quelle paire de nœuds appartenant au réseau. La stratégie de routage doit prendre en considération les changements de la topologie et des caractéristiques physiques du réseau (densité et hétérogénéité des nœuds), ainsi que la sécurisation (authentification, intégrité, confidentialité) des données circulant sur les nœuds intermédiaires.

Le projet SEREADMO étudiera un protocole de routage sécurisé et interopérable permettant de satisfaire les exigences des réseaux Ad Hoc, en exploitant la densité des nœuds pour d'une part distribuer la charge au niveau de chaque hôte mobile et pour d'autre part améliorer la sécurité et la sûreté de fonctionnement de tels réseaux, conditions indispensables à l'essor de cette technologie et des services associés.

Ce projet se situe dans le cadre de l'appel à propositions RNRT 2005, il est à caractère exploratoire et répond à l'objectif prioritaire intitulé « sécurité » tout en étant en lien avec les thèmes « mobilité et interopérabilité » ainsi que « intelligence ambiante ».

Le projet SEREADMO doit permettre de réaliser un protocole basé sur trois idées fondamentales :

[1] Le protocole utilise la transformée Mojette : la distribution des données ainsi que leur validité en intégrité sera définie par la mise en place dans le protocole de la transformée de Mojette, permettant le morcellement de l'information en multiples sous-ensembles redondants mais non sensibles unitairement.

[2] Le protocole utilise la technologie MIMO : la gestion des interférences et des liens radio asymétriques sera compensée par l'utilisation de cette technique de diffusion à antennes multiples.

[3] Le protocole dispose de mécanismes de sécurité permettant l'authentification, la confidentialité des trames de routage circulant dans les nœuds intermédiaires et les équipements finaux, ainsi qu'une architecture modulaire de sécurité des données sensibles : Ces mécanismes devront être dans les couches hautes OSI (Transport, Session) afin de pouvoir préserver l'interopérabilité du protocole sur différents types de médium (802.11x, Bluetooth, UWB,..), et seront mis en œuvre par une architecture matérielle adéquate (ex. TPM Trusted Platform Module, etc).

Tous ces points permettront d'étudier un protocole viable de routage ad hoc sécurisé sans contraintes majeures sur les équipements mobiles, tout en répondant au cahier des charges inhérent aux réseaux ad hoc.

4.2 Objectif de Recherche technologique

L'objectif de SEREADMO est de proposer un ensemble de standards (IETF/MANET) pour les protocoles de routage sécurisés dans les réseaux ad hoc.
Les protocoles réalisés seront testés et validés dans des environnements de simulation de type urbain (Adaptation NS2).
Par ailleurs, des démonstrateurs d'équipements seront réalisés, respectant une architecture permettant d'atteindre un haut niveau de sécurité. Ils permettront de valider les choix effectués dans un environnement urbain réel.

De plus, une étude des besoins de services sur les réseaux ad hoc sera menée. Celle-ci pourra être complétée ultérieurement par l'expérimentation de services concrets en environnement réel.

Les résultats de cette étude pourront être utilisés afin de mettre en œuvre des systèmes adaptés en premier lieu à l'ensemble des services de sécurité civile (les pompiers, la police, …) et des services de santé (transmission des données biométriques via les ambulances, systèmes de micro-capteurs, …), puis dans un second temps à l'ensemble de la population civile (Services d'informations routières, services urbains, régionaux, jeux, intelligence ambiante, …).

4.3 Relation avec les autres délivrables

Ce délivrable s'appuie sur la «spécification technique de l'architecture du protocole » (D12) et sur la « spécification du protocole de sécurité SEREADMO : SERANO » (D22) pour la définition et la réalisation de la plate-forme d'expérimentation.

4.4 Compatibilité avec les standards

Le protocole SERANO est compatible du protocole OLSR défini par la RFC 3626 [OLSR].

5. INTRODUCTION

Ce document décrit la plate-forme matérielle mise en œuvre et les expérimentations réalisées dans le cadre du projet RNRT SEREADMO 2005. Cette plate-forme de test doit permettre de valider les hypothèses de travail et les résultats de simulation effectués dans le cadre du projet sur le logiciel NS2. Dans cette optique, l'objectif de l'expérimentation est de démontrer les apports des hypothèses de travail du projet SEREADMO, c'est à dire :

- Prise en conte des contraintes de Mobilité (piétonne),
- Contrainte de topologie multi-chemins,
- Apport de la technologie MIMO dans le cadre Urbain,
- Capacité d'analyse du trafic et de la sécurisation mise en oeuvre par le protocole de routage

Pour cela, le chapitre 6 définit de manière globale la plate-forme utilisée dans le cadre de l'expérimentation. Le chapitre 7 détail la mise en œuvre de cette plate-forme et enfin le chapitre 8 décrit la méthode d'expérimentation et les différents scénarios mis en œuvre afin de valider le protocole et confronter les résultats de l'expérimentation aux résultats des simulations sous NS2.

6. DEFINITION DE LA PLATE-FORME

Ce chapitre décrit de manière globale la plate-forme mise en œuvre pour la validation du protocole SEREADMO. Pour cela, un premier paragraphe compare l'impact du protocole SEREADMO sur les différents nœuds du réseau par rapport au protocole OLSR. Un second présente le matériel mis en œuvre pour l'expérimentation, puis enfin chaque composant de la plate-forme est ensuite décrit dans un paragraphe spécifique.

6.1 Impact du protocole SEREADMO sur les nœuds du réseau

Le protocole SEREADMO étant basé sur le protocole OLSR, lui-même utilisant le protocole IP pour l'échange des données entre les nœuds du réseau, il est nécessaire afin de permettre le choix du matériel et du système d'exploitation de ces nœuds, de connaître l'impact logiciel de ces protocoles sur chacun des ces nœud. Pour cela, ce paragraphe rappel rapidement le fonctionnement d'une communication à l'aide du protocole IP, puis l'impact du protocole OLSR cette communication et afin celle du protocole SEREADMO.

6.1.1 Le protocole IP

Ce paragraphe présente de manière très simplifiée le fonctionnement d'une communication entre deux nœuds d'un réseau, Ad Hoc ou non, via le protocole IP. Ce paragraphe n'a pas pour but de décrire de manière complète le fonctionnement d'un réseau et du protocole IP, certaines notions seront donc

volontairement omises. Pour de plus amples informations sur ce protocole, le lecteur pourra consulter [TCPFOUND] et [TCPILLUSTR].

6.1.1.1 Réseau et sous réseau

Avant d'entrer de le détail de la communication entre deux nœuds d'un réseau, il est nécessaire de préciser ce que signifie la notion de réseau. Un réseau est un ensemble d'équipements communicants entre eux au travers d'un média de communication (câble, ondes radio, …). Lorsque que la composition d'un réseau devient trop importante, ou que des contraintes physiques ou d'administration ne permettent pas aux différents équipements d'utiliser le même média pour atteindre tous les nœuds, il est alors possible de décomposer ce réseau en plusieurs réseaux physiques plus petits (sous-réseaux) interconnectés entre eux par des équipements appelés passerelles (ou routeurs suivant leurs complexités).

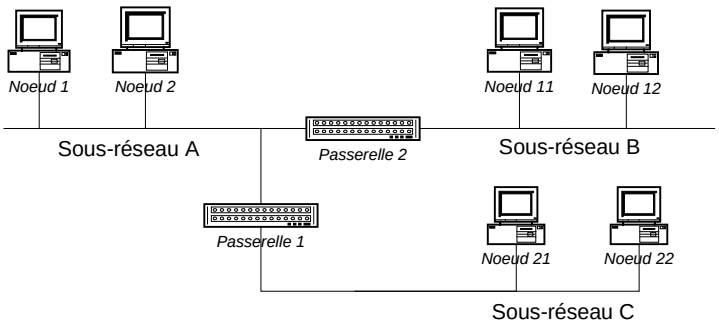

Figure 1 : Exemple d'interconnexion de sous-réseaux

Sur la figure ci-dessus, trois sous-réseaux : A, B et C sont interconnectés au moyen de deux passerelles : 1 et 2. Lorsque les nœuds 1 et 2 présents sur le même sous-réseau A, souhaite communiquer entre eux, ils s'échangent directement les données sans utiliser de passerelle. La communication est du type 1 saut, car les données vont directement du nœud 1 au nœud 2. A l'inverse, lorsque le nœud 1 du sous-réseau A souhaite communiquer avec le nœud 12 du sous-réseau B, il doit obligatoirement envoyer toutes ses données à la passerelle 2, qui se chargera de les retransmettre au nœud demandé. Il n'y a donc pas de communication directe entre les nœuds de sous réseaux différents. Dans ce cas, la communication nécessite plusieurs sauts : un saut du nœud 1 à la passerelle 2 et un saut de la passerelle au nœud 12.

Dans le cas d'un réseau Ad Hoc, chaque nœud du réseau ne peut physiquement communiquer qu'avec ses proches voisins (voisin à 1 saut). On peut donc faire une analogie entre un nœud et ses voisins à 1 saut et un sous-réseau. Lorsqu'un nœud souhaite communiquer avec un autre non présent parmi ses voisins à 1 saut, il est doit donc obligatoirement faire transiter ses données par l'un de ses voisins qui se chargera à son tour les transmettre en direction du destinataire.

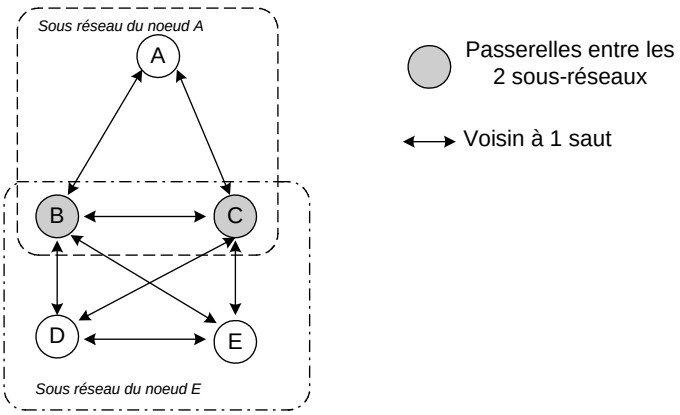

Figure 2 : Exemple de sous-réseaux Ad Hoc

Sur la figure ci-dessus, lorsque le nœud A souhaite communiquer avec le nœud E, il doit utiliser les nœuds B ou C en tant que passerelle.
Dans les deux exemples ci-dessus, le choix de la passerelle, 1 ou 2 pour la Figure 1 et B ou C pour la Figure 2, est réalisé par le protocole IP, au moment de l'envoi des données utilisateur, à l'aide d'informations contenues dans une table dite de routage.

6.1.1.2 Table de routage

La table de routage est une table de correspondance entre un nœud destinataire et le nœud suivant (1 saut) auquel le nœud local doit envoyer ses données.

Dans le cas de la Figure 1, la table de routage du nœud 1 serait du type :

Destinataire	Passerelle
11	2
12	2
21	1
22	1

Cette table est mise à jour selon deux méthodes :

- Soit manuellement, l'administrateur du réseau est alors en charge de configurer chaque équipement,
- Soit automatiquement, des outils et protocoles de mise à jour, tel que OLSR, sont alors utilisés.

6.1.2 Le protocole OLSR

La communication au sein d'un réseau Ad Hoc à l'aide du protocole OLSR est décomposée en deux étapes :

1. La détermination des chemins entre les nœuds
2. Le transfert des données utilisateur

6.1.2.1 Détermination des chemins

Dans la première étape, la détermination des chemins entre les nœuds du réseau, le protocole OLSR est implémenté au sein d'une application de type utilisateur (donc sans modification du système d'exploitation). Ces applications, présentes sur les différents nœuds du réseau, s'échangent les messages définis par le protocole OLSR (voir délivrable D11) afin de découvrir la topologie du réseau.

Après détermination de cette topologie et du calcul des chemins, le protocole OLSR réalise une mise à jour de la table de routage IP.

Dans le cas de la Figure 2, la table de routage IP du nœud A pourrait être la suivante :

Destination	Passerelle
E	B
D	B

Dans cet exemple, bien que les chemins A=>C=>E, et A=>C=>D existent, ils ne seront pas ajoutés par OLSR à la table de routage IP, et ne seront donc pas exploités.

6.1.2.2 Transfert des données utilisateur

Lors de l'étape du transfert des données utilisateur, le protocole OLSR ne réalise aucune opération. Le transfert est entièrement réalisé par le protocole IP sur la base du contenu de sa table de routage.

Le protocole OLSR est donc limité à la mise à jour de la table de routage IP réalisée lors de l'étape 1 (détermination des chemins) et n'intervient pas dans le transfert des données utilisateur entre les nœuds. Ce

protocole peut donc être vu comme un simple outil de configuration automatique de la table de routage IP des nœuds du réseau en fonction de la topologie de ce dernier.

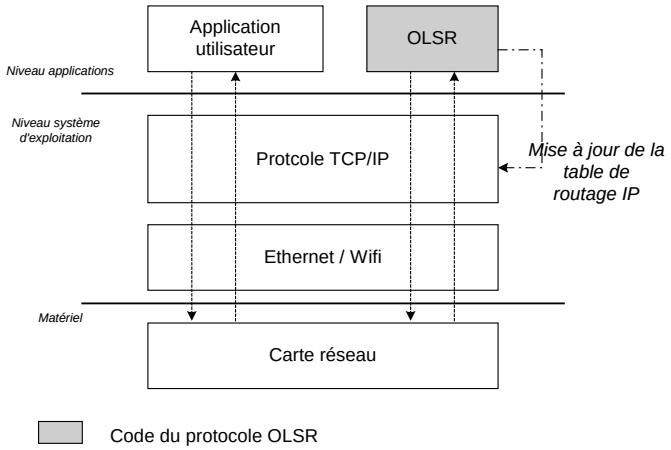

Figure 3 : Impact du protocole OLSR sur les nœuds du réseau

L'utilisation du protocole OLSR, n'a donc aucun impact sur le fonctionnement interne du système d'exploitation des différents nœuds du réseau. Il est donc facilement utilisable sur n'importe quel type de système, sous condition que ce dernier fournisse des API de mise à jour de la table de routage IP.

Ce faible impact d'OLSR sur les différents nœuds du réseau, à cependant une contrepartie importante. En effet, pour chaque route calculée à partie des données topologiques, OLSR ne peut indiquer que le premier nœud intermédiaire (voisin à 1 saut) dans la table de routage IP. De plus, lors d'un transfert de données, chacun des nœuds participant au transfert n'a plus accès à la route initialement prévue par le nœud émetteur. Le protocole IP de chacun de ces nœuds est alors obligé de se baser uniquement sur le contenu de sa propre table de routage pour poursuivre le transfert. Le chemin initial peut donc être modifié au fil du transfert par les nœuds intermédiaires.

6.1.3 Le protocole SEREADMO

Afin de remplir tous les objectifs fixés par le projet, le protocole SEREADMO intervient à divers endroits dans la communication entre les nœuds.

Comme pour les communications à l'aide du protocole OLSR, les communications à l'aide du protocole SEREADMO sont décomposées en deux phases :

1. La détermination des chemins entre les nœuds
2. Le transfert des données utilisateur

6.1.3.1 Détermination des chemins

Pour déterminer les chemins entre les nœuds du réseau, le protocole SEREADMO est implémenté de manière identique au protocole OLSR. Il est donc réalisé à l'aide d'une application de type utilisateur. Cependant contrairement au protocole OLSR, après la détermination de la topologie du réseau, l'application ne réalise pas de mise à jour de la table de routage IP. Cette fonction étant devenue inutile, comme le montreront les paragraphes ci-dessous.

6.1.3.2 Transfert des données utilisateur

Contrairement au protocole OLSR, qui laisse le soin au protocole IP de transférer les données utilisateur, le protocole SEREADMO intervient dans ce transfert. Cet impact est réalisé en 3 points différents :

1. La décomposition de toutes les données utilisateur à l'aide de l'opérateur de projection Mojette.

2. Le choix d'un chemin différent et définitif pour chaque projection Mojette.

3. Le cryptage de toutes les données issues des nœuds à l'aide du protocole SERANO

Le protocole SEREADMO doit donc court-circuiter le fonctionnement standard du protocole IP, pour qui tous les paquets de données à destination d'un même nœud sont envoyés à la même passerelle, donc au même nœud voisin dans le cas d'un réseau Ad Hoc. Et, contrairement au protocole OLSR, ne laisse plus le soin au protocole IP de mettre à jour les chemins au fil des nœuds intermédiaire traversés.

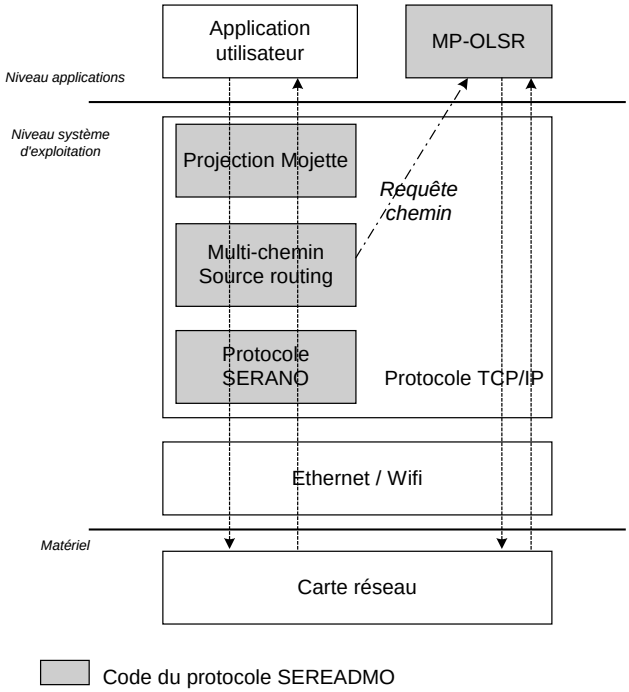

Figure 4 : Impact du protocole SEREADMO sur les nœuds du réseau

Certaines fonctions du protocole IP sont donc remplacées par un accès aux données topologiques générées par le protocole lors de la phase de détermination des chemins (phase 1) et par un accès aux chemins contenus dans les trames envoyées.

6.1.4 Conclusion

Comme le précise les paragraphes ci-dessus, le protocole SEREADMO impact fortement le fonctionnement de la pile TCP/IP contrairement au protocole OLSR, qui lui ne modifie pas cette dernière. L'implémentation de ce nouveau protocole nécessite donc de pouvoir modifier le comportement par défaut du système d'exploitation installé sur les différents nœuds du réseau. Ce système d'exploitation retenu pour les nœuds de la plate-forme d'expérimentation doit donc être facilement modifiable et adaptable.

6.2 Description de la plate-forme

Le but de la plate-forme mise en œuvre est de valider les apports du protocole SEREADMO par rapport au protocole OLSR et aux autres protocoles de routage pour réseau Ad hoc. Pour cela, elle est donc composée d'équipements mobiles utilisés pour former les nœuds d'un réseau Ad Hoc. Mais, afin de valider le comportement de ces nœuds, deux équipements supplémentaires sont également utilisés. Le premier est une machine d'analyse qui collectera en temps réel des informations sur les nœuds du réseau, dans le but de les comparer à celles obtenues sous NS2 et de permettre de faire évoluer en temps réel les différents scénarios de test, afin d'adapter et optimiser l'expérimentation. Le second équipement est un nœud espion, du même type que les nœuds du réseau, mais cette fois, utilisé pour valider l'aspect sécurité du protocole.

La plate-forme matérielle est donc composée des équipements suivants :

- **Les nœuds du réseau Ad Hoc :** 10 équipements mobiles ayant une capacité de communication sans fils adaptée (Wifi 802.11a/n), une possibilité d'accélération cryptographique (présence d'une TPM) et un mode de communication filaire afin de connecter ces nœuds à l'équipement d'analyse. Cette voie filaire étant la plus adaptée pour transmettre des informations en temps réels sans perturber les communications radios du réseau Ad hoc, et donc sans risquer de fausser l'expérimentation.

- **La machine d'analyse :** Cet équipement de type ordinateur fixe a une capacité de traitement adaptée aux opérations d'analyse des données récoltées sur les nœuds du réseau et possède un réseau filaire longue portée (env 2 km) de type RS485 ou Ethernet permettant la liaison de diagnostic et d'analyse in-situ des nœuds du réseau.

- **Le nœud espion :** Ce nœud également mobile, dispose d'une capacité de communication sans fils et d'outils spécifiques permettant la récupération et l'injection de trames dans un réseau. Il est utilisé afin de valider l'aspect sécurité du protocole, en permettant l'interception des données échangées entres les nœuds et en envoyant des données arbitraires dans un but de perturbation du réseau.

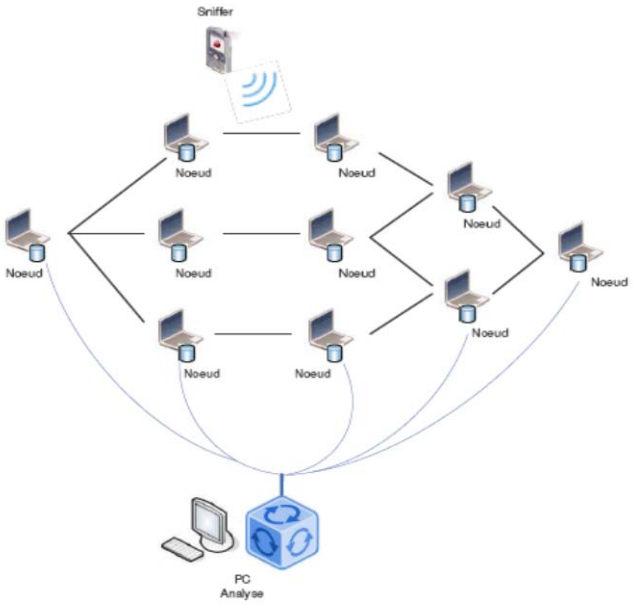

Figure 5 : Exemple de mise en œuvre de la plate-forme

Le coût des nœuds du réseau et la logistique nécessaire durant l'expérimentation (mobilité des nœuds), limite à 10 équipements le réseau mise en œuvre. Cette plate-forme, bien que limitée en nombre de nœuds, permet cependant de valider toutes les spécificités du protocole, telles que la mobilité des nœuds, le routage multi-chemins et la sécurisation du trafic, à l'exception de l'utilisation du protocole dans un réseau dense, qui nécessiterait un minimum de 50 à 100 nœuds.

Le nombre de nœuds ainsi retenu permet de valider le protocole sur des configurations de réseaux mobiles standards, tel que celui de la Figure 6.

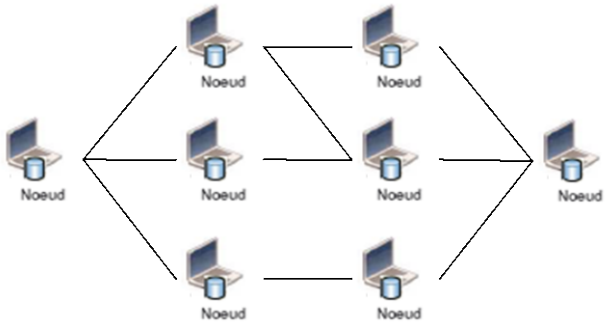

Figure 6 : Configuration standard de la plate-forme

Ainsi que sur des configurations atypiques telles que celle de la Figure 7, qui en maximisant le nombre de sauts entre les nœuds extrêmes permet de valider la capacité de source routing du protocole.

Figure 7 : Maximisation du nombre de sauts

Ou encore, une configuration telle que celle de la Figure 8, qui en maximisant le nombre de chemins possibles entre les nœuds extrêmes, permet de valider la capacité multi-chemins du protocole.

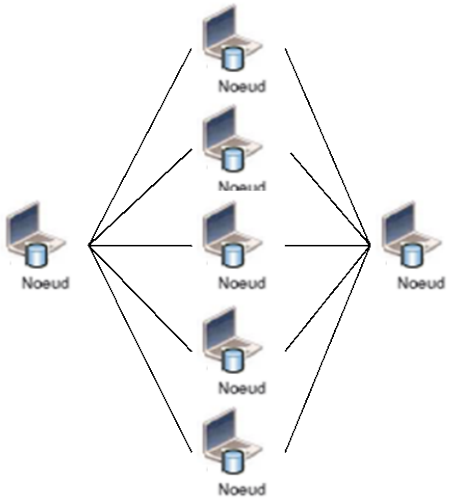

Figure 8 : Maximisation du nombre de chemins

Les prochains paragraphes vont maintenant détailler les matériels et logiciels (système d'exploitation et outils) utilisés pour réaliser des équipements de cette plate-forme.

6.3 Les nœuds du réseau

Les choix réalisés pour la réalisation de la plate-forme sont basés sur un besoin d'expérimentation / validation et non d'intégration / industrialisation du protocole SEREADMO. Ces choix prennent donc principalement en compte des outils matériels et logiciels permettant une grande souplesse de modifications et d'instrumentalisation.

6.3.1 Plate-forme matérielle

Le protocole SEREADMO étant axé en partie sur la mobilité des nœuds du réseau, les équipements utilisés pour ces nœuds doivent donc pourvoir être mobiles. Depuis quelques années, le choix en outils mobiles c'est très largement développé. De nombreux types d'équipements sont disponibles avec des capacités de stockage et de traitement extrêmement variés. Ces différentes machines peuvent être regroupées en grandes catégories :

- Téléphone cellulaires (Smart phone)
- PDA,
- Tablet PC
- Ordinateur portable

Contrairement à une possible phase d'industrialisation, où un maximum de matériel mobile compatible devra être envisagé, la phase d'expérimentation, nécessite plutôt du matériel dont l'ajout de composants (nouvelles cartes réseau Wifi, puce TPM) est facilement réalisable et où les modifications du système d'exploitation et l'ajout de nouvelles applications reste le plus simple et le plus accessible possible. Le choix du matériel pour les nœuds du réseau Ad Hoc s'est donc porté sur des machines de type ordinateur portable.

Les nœuds du réseau seront donc des ordinateurs portables, type PC, possédant les caractéristiques suivantes :

- Carte réseau Wifi a/g
- Carte réseau Ethernet (pour la connexion à la machine d'analyse)
- Des ports d'extension (USB, PCMCIA, …) permettent l'ajout de nouveaux périphériques tels que des cartes réseau Wifi, des puces de cryptages, …

Ce type d'équipement permet en effet l'ajout de nouveaux composants matériels, autorise la communication sur différents canaux (Ethernet et Wifi), et fourni un grand choix d'outils et de systèmes d'exploitations répondants aux besoins d'implémentation du protocole, tel que décrit au paragraphe 6.1.3.

6.3.2 Plate-forme logicielle

6.3.2.1 Système d'exploitation

Sur le matériel retenu pour les nœuds du réseau, décrit au paragraphe précédent, deux types de système d'exploitation sont majoritairement disponibles :

- Les systèmes de type « Microsoft Windows » : Windows XP ou Windows Vista, …
- Les systèmes de type « Unix » : Linux, BSD, Sun OS, …

Comme indiqué dans le paragraphe précédent, pour la phase d'expérimentation, le matériel et le système d'exploitation des nœuds du réseau doivent être facilement modifiable. Il est donc nécessaire d'avoir :

- Soit accès aux sources du système,
- Soit avoir des API suffisamment évoluées pour permettre l'implémentation de toutes les fonctions du nouveau protocole.

Pour les 2 types de système d'exploitation définis ci-dessus, l'utilisation des API n'est pas suffisante pour l'implémentation de toutes les fonctions proposées par le protocole SEREADMO ([INS200] [WININT], [HERBERT]), un accès aux sources du système à donc était un élément décisif pour le choix du système d'exploitation retenu.

Ce choix s'est tourné vers les systèmes d'exploitation de type Linux, pour leur grande facilité d'accès au code source et la communauté importante de développeur.

Les nœuds composants le réseau de la plate-forme seront donc des ordinateurs de type PC portable, fonctionnant sous Linux.

6.3.2.2 Implémentation d'OLSR

Afin de ne pas re-coder l'ensemble des fonctions du protocole OLSR intégrées au protocole SEREADMO, une implémentation existante d'OLSR est utilisée.

La liste ci-dessous présente les principales implémentations d'OLSR hors implémentations dédiées uniquement aux outils de simulation, tels que NS2, donc non utilisables pour la plate-forme d'expérimentation.

- MeshTool, OWR200 : Suite commerciale d'outils de routage intégrant OLSR de la société Luceor [LUCEOR].
- 6WINDGate : Suite commerciale d'outils de routage intégrant OLSR de la société 6Wind [6WIND].
- OLSRD : Implémentation réalisée par Andreas Tønnesen durant sa thèse [OLSRD], [TONNESEN]. Cette implémentation conforme à la RFC 3626, fonctionne sur de nombreuse plate-forme (Windows, Mac OSX, linux, Open/FreeBSD, …).La principale faiblesse de cette implémentation est son manque de performance lors de son utilisation sur de grands réseaux (plusieurs centaines de nœuds). Ce défaut à conduit à la mise en place d'un nouveau projet OLSR-NG.
- OLSR-NG : Implémentation encore à l'état de projet vise à combler les faiblesses Olsrd [OLSRNG].
- OLSR v3 : Ancienne implémentation de l'INRIA [OLSRV3]. Cette implémentation est basée sur une version draft v3 d'OLSR, elle n'est donc pas entièrement conforme à la RFC 3626.
- OOLSR : Implémentation de référence de l'INRIA [OOLSR]. Cette implémentation est conforme à la RFC 3626.
- PyOLSR : Implémentation en Python de l'INRIA [PYOLSR]. Cette implémentation n'est plus maintenue et a été remplacée par OOLSR.
- NRL-OLSR : Implémentation réalisée par l'US Naval Research Laboratory (NRL) [NROLSR], basée sur la RFC 3626 à la quel certaines fonctions ont été ajoutées.
- CRC-OLSR : Implémentation réalisée par le Centre de recherche sur les communications Canada (CRC) [CRCOLSR]. Cette version est basée sur l'implémentation OLSR v3 de l'INRIA et a été adaptée afin de répondre à la RFC 3626.
- QOLSR / Qolyester : Projet de mise en œuvre de QoS (Quality of service) du Laboratoire de Recherche en Informatique (LRI), Université Paris Sud 11 [QOLSR]. Cette implémentation répondre à la RFC 3626. Pour le moment, seul l'implémentation répondant à la RFC 3626 est disponible. La partie QoS n'étant pas encore à disposition.
- NOA-OLSR : Implémentation de l'INRIA, basée sur OOLSR, permettant l'auto-configuration du réseau (attribution d'adresses IP aux nœuds, …) [NOAOLSR].
- SMOLSR-MOLSR : Implémentation de l'INRIA, basée sur OOLSR, permettant la diffusion multicast des données utilisateur [SMOLSR].
- University of Valencia - OLSR : Implémentation de l'université polytechnique de Valence (Espagne) [UPVA]. Cette implémentation est un portage d'OLSR v3 sur plate-forme Microsoft Windows (2000 et Pocket PC) à l'aide d'une librairie interne : PICA.

Pour des raisons de facilité d'accès au code source et de modification de celui-ci, seules les implémentations open-sources ont été retenues dans le cadre de l'expérimentation, ainsi que celles répondant à la RFC 3626 et n'apportant pas de fonctionnalités spécifiques telles que le multicaste ou QoS.

Le tableau ci-dessous résume les principales implémentations retenues et leurs caractéristiques.

Nom	Dernière version	Langage	Système d'exploitation	Remarques
Olsrd	0.5.5 02/2008	C	Linux, Windows, I-phone, Mac OSX, …	Utilisée sur de nombreux sites et régulièrement mis à jour
Olsr-NG	Na	Na	Na	Non encore disponible
PyOLSR	0.0.2 2003	Python	Linux, Windows	N'est plus supporté
OOlsr	0.99.15 11/2004	C++	Linux, Windows	
NRL-Olsr	7.8.1 08/2007	C++	Linux, Windows	
CRC-Olsr	2004	C	Linux	
Qolyster	06/2008	C++	Linux	
Univ. Valencia	10/2002	C	Linux, Windows	

Parmi toutes ces implémentations, le choix s'est tourné vers Olsrd, car cette implémentation est toujours maintenue et fait l'objet de corrections et d'évolutions (optimisations, portage sur d'autres plate-formes, …). Elle est de plus exploitée dans de nombreuses villes, notamment au travers d'une version linux dédiée au matériel embarqué : OpenWrt [OPENWRT], comme par exemple à Bruxelles [RESCIT], Lille [LILLESF], Nantes [NANTESW], Caen [CAENSF]… rattachées pour certaines à la fédération France Wireless [FRAWLS].

6.4 La machine d'analyse

Le but de la machine d'analyse est de collecter en temps réel un maximum d'informations sur le fonctionnement des nœuds du réseau, c'est à dire le calcul des routes (nombre de routes, nœuds visibles,…) et sur les données utilisateur transférées (nombre de paquets échangés, nombre d'échecs de transfert, …).

Cette machine d'analyse, contrairement aux nœuds du réseau, n'a pas besoin d'être mobile, ni d'avoir des capacités de communication sans fils. Par-contre, elle doit être capable de traiter (recevoir, stocker et afficher) en temps réel toutes les données envoyées par les nœuds du réseau, via leur moyen de communication filaire. Seul l'aspect capacité de stockage et puissance de traitement à donc été pris en compte et non l'aspect ajout de composants matériels, modifications du système d'exploitation.

Le matériel retenu est donc du type PC serveur fonctionnant sous Linux.

6.5 Le nœud espion

Cet équipement utilisé pour toute la partie sécurité de l'expérimentation, doit pouvoir supporter les mêmes contraintes que les nœuds du réseau, c'est à dire évolution du matériel, mobilité, mise en œuvre du protocole SEREADMO. Il est donc du type ordinateur portable et utilise un système d'exploitation de type Linux. La distribution Linux retenu est la distribution Backtrack [BACK], dédiée à l'analyse et l'attaque de réseaux sans fils.

En effet, cette distribution Linux est composée d'outils pour :

- Capter tous les paquets (données utilisateurs ou routage) en transit entre ses proches voisins.
- Rejouer certains paquets (données utilisateurs ou routage)
- Générer des paquets aléatoires afin perturber le trafic

Ces différents outils permettront de réaliser les différents attaques décrites dans le délivrable D21, Partie 2 (Cibles de sécurité contexte Ad hoc) et donc de valider que les spécifications du protocole SEREADMO définis dans les délivrables D12 (Spécification technique de l'architecture du protocole) et D22 (Spécification du protocole de sécurité SEREADMO : SERANO) permettent de garantir une sécurisation des données utilisateur.

7. REALISATION DE LA PLATE-FORME

Après une présentation globale de la plate-forme d'expérimentation au chapitre précédent, ce chapitre décrit plus en détail la réalisation de cette plate-forme. Chaque type d'équipement composant cette plate-forme est présenté dans un paragraphe spécifique qui détail les différents travaux réalisés.

7.1 Les nœuds du réseau

Comme décrit dans le chapitre précédent, les nœuds du réseau sont composés d'ordinateurs de type PC portable fonctionnant sous Linux.

Les travaux réalisés sur ces nœuds sont les suivants :

1. Implémentation du protocole MP-OLSR sous forme d'une application utilisateur, basée sur l'application Olsrd qui implémente le protocole OLSR

2. Modification de la partie haute de la pile TCP/IP pour décomposer tous les paquets de données en provenance des applications utilisateur à l'aide de l'opérateur de projection Mojette

3. Modification de la fonction de routage du protocole IP pour prendre en compte les différents chemins obtenus à l'aide de la fonction de calcul des routes du protocole MP-OLSR.

4. Modification de la partie basse de la pile TCP/IP afin de crypter tous les paquets de données à l'aide du protocole SERANO

Les prochains paragraphes vont maintenant détailler les différents travaux réalisés.

7.1.1 Implémentation du protocole MP-OLSR

Comme indiqué au paragraphe 6.3.2.2, l'implémentation du protocole MP-OLSR, se base sur une implémentation déjà existante du protocole OLSR, nommée Olsrd. Par rapport au fonctionnement standard du protocole OLSR, les modifications suivantes sont réalisées sur l'implémentation fournie par Olsrd.

1. Suppression du calcul des routes en fonction des modifications de la topologie du réseau. Les routes initialement calculées par le protocole OLSR ne comportent qu'un seul trajet pour le transfert de données entre 2 nœuds. Il est donc nécessaire de modifier l'algorithme utilisé afin de prendre en compte les spécificités du protocole MP-OLSR (multi-chemins). De plus, afin de limiter les calculs réalisés par le protocole, les différents chemins proposés par MP-OLSR, ne sont pas calculés dès détection d'une modification de la topologie, mais en fonction des besoins lors du transfert des données utilisateur, comme le prévois la définition du protocole (voir délivrable D12).

2. Ajout d'une fonction de réception des demandes de chemins envoyées par la pile TCP/IP pour le transfert des données utilisateur

3. Ajout de la fonction de calcul des multi-chemins.

Le diagramme (Figure 9) ci-dessous présente les différentes opérations réalisées par le protocole OLSR. A chaque réception d'une trame HELLO ou TC les données topologiques sont mises à jour si nécessaire. En cas de modification de ces dernières, la totalité des routes sont recalculées et la table de routage IP est mise à jour, afin d'indiquer pour chaque nœud du réseau, le nœud voisin du nœud courant à utiliser en tant que passerelle pour l'envoi des données utilisateur.

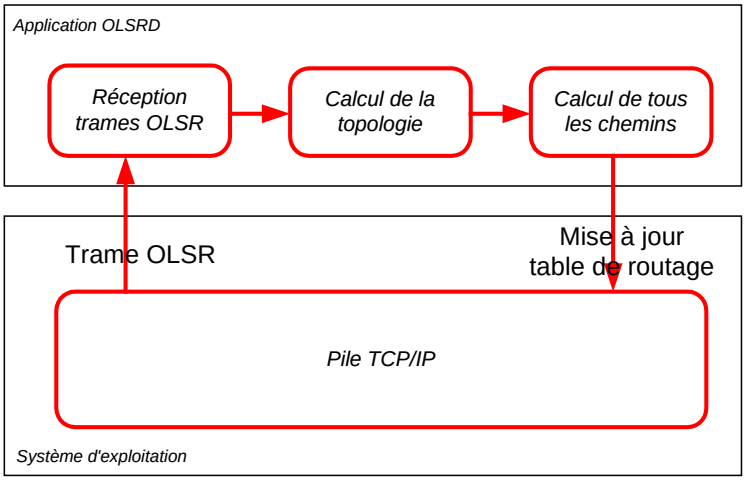

Figure 9 : Opérations réalisées par le protocole OLSR

Le diagramme ci-dessous (Figure 10) présente les opérations réalisées par le protocole MP-OLSR. Lors de la réception des messages HELLO et TC, comme dans le cas du protocole OLSR, les données topologiques sont mises à jour. Cependant, après mise à jour de ces données, aucune autre opération n'est réalisée.
Lors de l'envoi de données utilisateur, la pile TCP/IP lance une requête à l'application MP-OLSR, qui :

- en cas de modification des données topologiques, ou lors de la première demande de chemin, calcul l'ensemble des chemins possibles entre le nœud courant et le nœud destinataire, puis retourne l'un de ces chemins.
- lors des appels suivants, retourne simplement les autres chemins précédemment calculés.

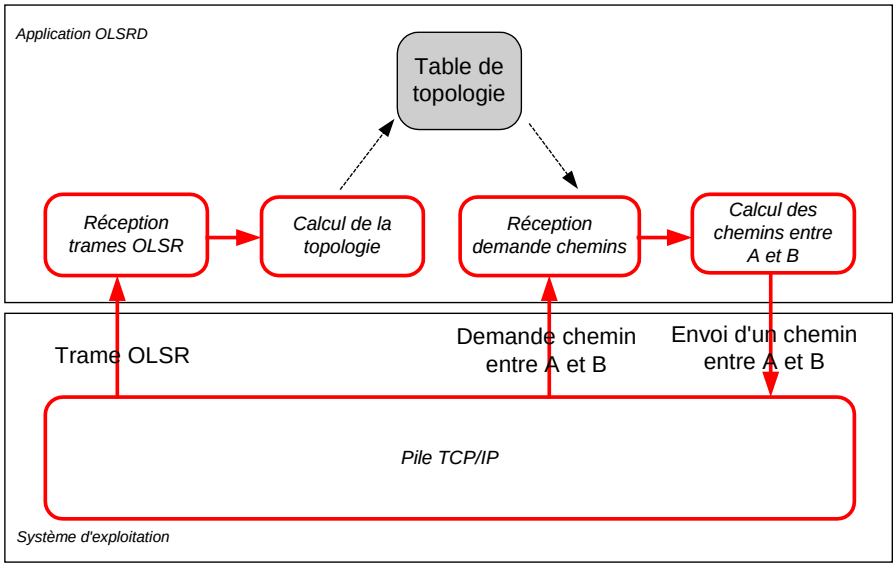

Figure 10 : Opérations réalisées par le protocole MP-OLSR

Pour implémenter ces fonctions, toute la partie de gestion des trames OLSR (TC et HELLO) est issue de l'application Olsrd. La partie réception des demandes de chemins et les calculs des différents chemins est ajoutée également à cette application. L'envoi des demandes depuis la pile TCP/IP est traité dans les paragraphes ci-dessous et se base sur les capacités de Linux d'envoyer depuis le cœur du système d'exploitation des requêtes (signaux) et de permettre aux applications de transmettre des données à ce système (appels IOCTL).

7.1.2 Implémentation de la projection Mojette

Comme le prévoit le protocole SEREADMO, toutes les données utilisateur envoyées au travers du réseau Ad Hoc sont décomposées à l'aide de l'opérateur de projection Mojette, afin d'être transmises au nœud destinataire via des chemins différents.
Pour réaliser cette projection, la partie haute de la pile TCP/IP, c'est à dire les fonctions sockets utilisées par les applications utilisateur, est modifiée afin d'y inclure la fonction de projection.
Vis à vis du reste de la pile TCP/IP, les différentes projections ainsi générées seront donc vues comme des données directement générées par l'utilisateur.

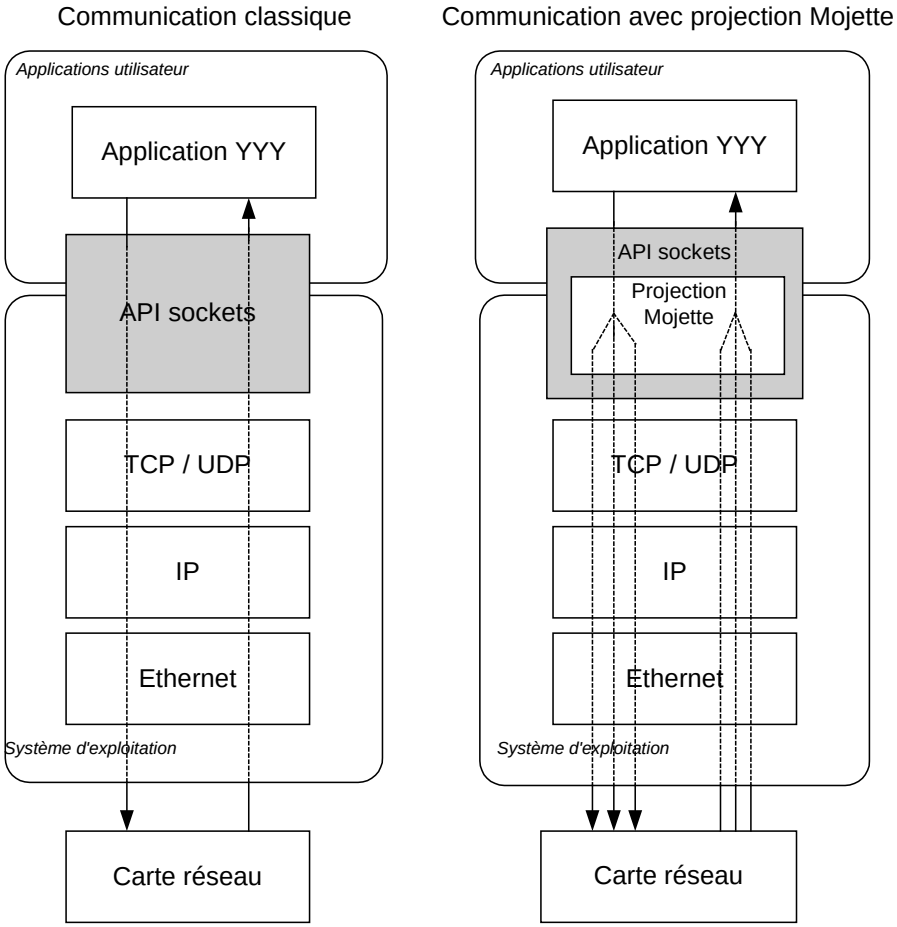

Figure 11 : Principe de décomposition du flux des données utilisateur

Afin de limiter l'impact sur le système d'exploitation et faciliter les mises à jour, les modifications réalisées utilisent les capacités de Linux à remplacer durant son exécution, à l'aide de libraires dynamiques et de la fonction LD_PRELOAD, certaines parties de son implémentation.

L'utilisation de ces librairies dynamiques permettra lors de la phase de validation terrain d'activer ou non l'opérateur de projection Mojette, sans l'arrêt ou la réinstallation du système d'exploitation, facilitant ainsi la mise au point.

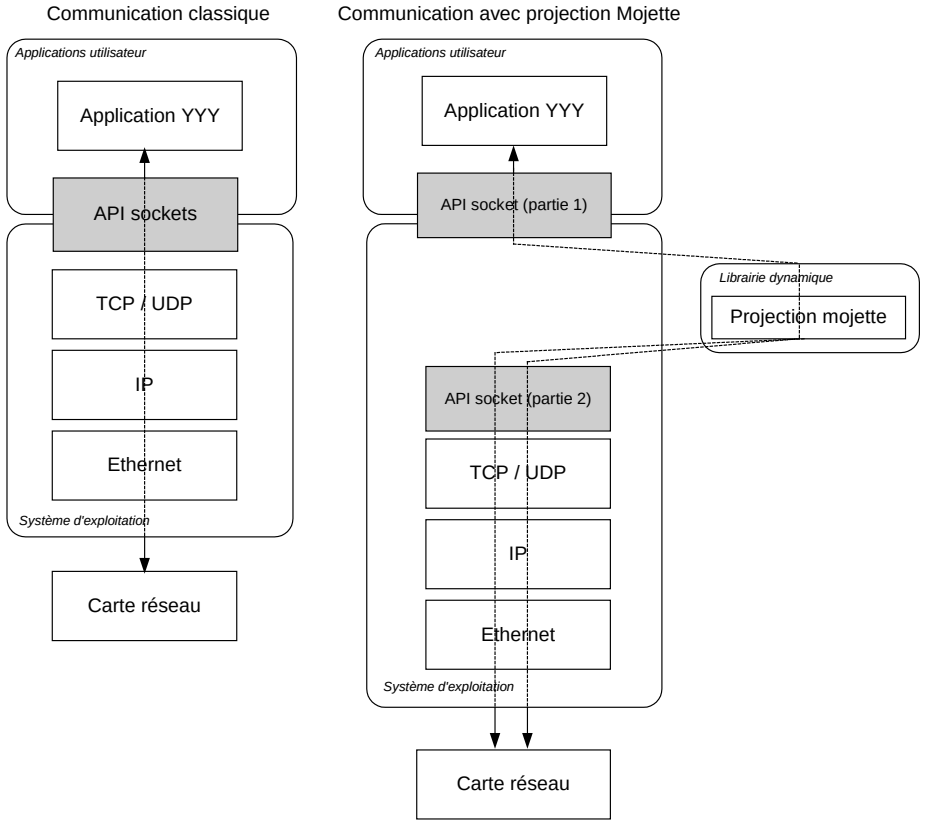

Figure 12 : Mise en oeuvre de la projection Mojette

Cette librairie dynamique détourne tous les appels aux fonctions sockets, de la pile TCP/IP et des applications utilisateur, pour l'envoi ou la réception des données. Après projection, ou reconstruction (pour la partie réception), les fonctions originales détournées sont rappelées avec le résultat de l'opérateur Mojette. Il n'y aura donc pas de modifications nécessaires de la pile TCP/IP, ni des applications utilisateur pour l'ajout de cet opérateur de projection.

7.1.3 Implémentation du routage des données utilisateur

7.1.3.1 Introduction

Comme décrit au paragraphe 6.1.2, le protocole OLSR n'intervient pas dans le transfert des données utilisateur, qui est reste à la charge du système d'exploitation, via le protocole IP et les tables de routage. Bien qu'un chemin soit dont prévu par le protocole OLSR, depuis le nœud initial jusqu'au destinataire, celui-ci peut donc continuellement modifié par les systèmes d'exploitation des nœuds traversés en fonction de leurs propres tables de routage. A l'inverse, le protocole SEREADMO utilisant le source routing, intervient de manière très précise sur le transfert de ces données, en forçant le chemin à emprunter sur les différents nœuds du réseau. Il devient donc nécessaire de modifier le comportement par défaut des fonctions de transferts des différents nœuds du réseau.

7.1.3.2 Transfert avec OLSR

Lors du transfert des données utilisateur dans un réseau IP, l'envoi des trames est réalisé à l'aide des tables de routage IP, comme le montre la Figure 13. Ce routage des trames par le protocole IP est décomposé en deux étapes :

- **Emission d'une trame utilisateur :** lors de la réception d'une trame en provenance des couches TCP / UDP, si elle-ci n'est pas à destination d'un voisin à 1 saut, le protocole IP recherche dans sa table de routage l'adresse du nœud utilisé en tant que passerelle vers le destinataire.

- **Réception d'une trame de la couche physique :** lors de la réception d'une trame de la couche physique, le protocole IP transfert la trame aux couches TCP/UDP si le nœud courant est le destinataire du paquet, ou renvoi la trame dans le circuit d'émission s'il le nœud courant n'est pas le destinataire. Il y donc à nouveau recherche du nœud voisin utilisé en tant que passerelle

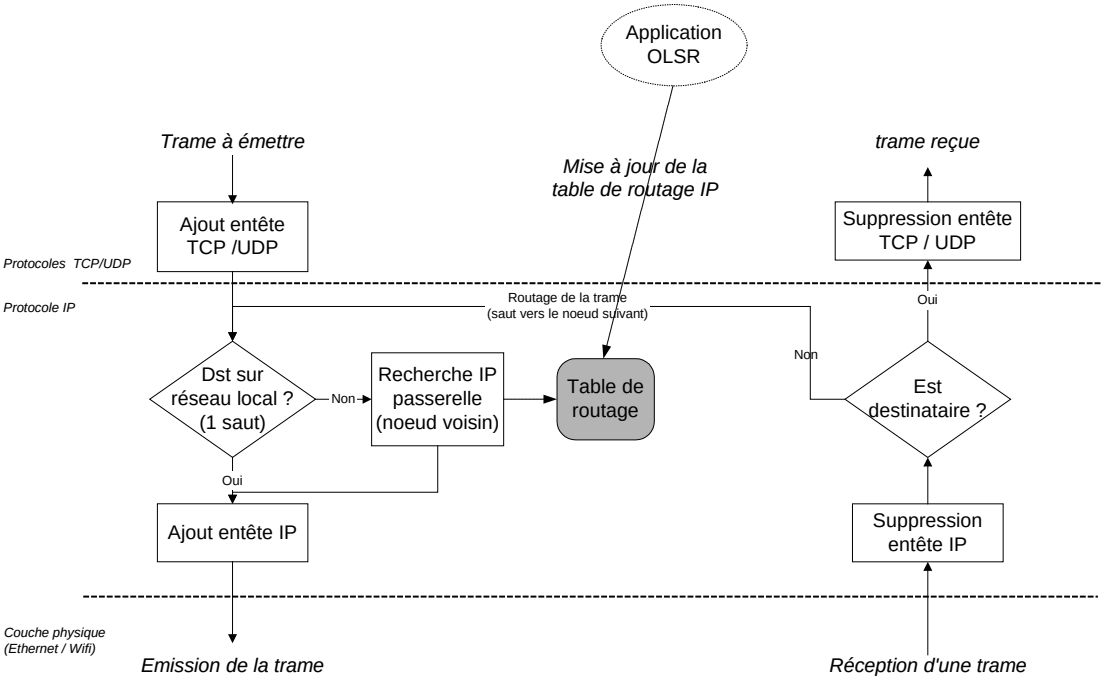

Figure 13 : Routage des données sur un nœud OLSR

7.1.3.3 Transfert avec SEREADMO

Dans le cas du protocole SEREADMO, on rencontre donc les problèmes suivant avec cette implémentation :

1. Chaque nœud présent sur le chemin consulte sa propre table de routage pour poursuivre le transfert. Ces nœuds n'utilisent donc pas le chemin ajouté au contenu de la trame.

2. Lors de l'émission de 2 trames consécutives vers le même destinataire, si la table de routage n'a pas était mise à jour immédiatement après l'envoi de la première trame, le nœud émetteur obtiendra de sa table de routage le même voisin utilisé en temps que passerelle vers le destinataire. Il y a donc perte de la capacité d'émission multi-chemins.

Afin de palier aux problèmes induits par ce fonctionnement de la couche IP, cette dernière est modifiée afin de correspondre au synoptique de la Figure 14. C'est à dire :

- **Emission d'une trame utilisateur :** lors de la réception d'une trame en provenance des couches TCP / UDP. La nouvelle fonction de routage interroge l'application MP-OLSR de découverte de la topologie. Cette dernière transmet à la couche IP le chemin (nœuds à parcourir). La nouvelle fonction de routage ajoute alors ce chemin au contenu de la trame et l'envoi au premier nœud de celui-ci (qui est obligatoirement un nœud voisin)

- **Réception d'une trame de la couche physique :** lors de la réception d'une trame de la couche physique, le protocole IP transfert la trame aux couches TCP/UDP si le nœud courant est le destinataire du paquet, ou recherche le prochain nœud sur le chemin à l'aide du chemin complet présent dans le contenu de la trame reçue et transfert celle-ci à ce nœud.

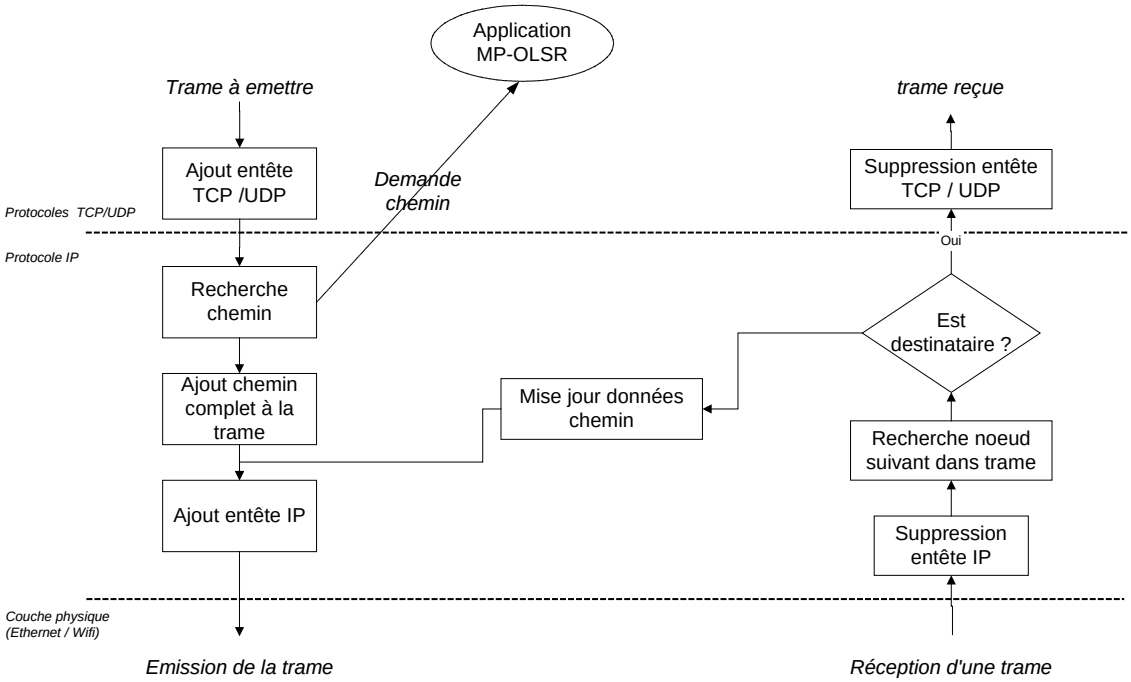

Figure 14 : Routage des données sur un nœud SEREADMO

Durant l'étape de réception des trames afin de prendre en compte la capacité du protocole, dans le cas non sécurisé, à réagir face à une perte de chemin (prochain nœud du chemin non accessible par le nœud courant). La nouvelle fonction de routage interroge l'application MP-OLSR, comme lors d'une demande de chemin, afin de valider à l'aide des données topologiques la présence du prochain nœud du chemin parmi les voisins à 1 saut et dans le cas contraire d'obtenir un nouveau chemin.

L'implémentation de ces nouvelles fonctions de routage nécessite, pour la partie de demande des chemins, de modifier la pile TCP/IP. Cependant, l'implémentation du source routing peut elle, être réalisée de 2 manières différentes :

1. En utilisant les champs « options » du protocole IP destinés au source routing,

2. En modifiant le fonctionnement de la pile TCP/IP pour ajouter une fonction de source routing dédiée au protocole SEREADMO.

Bien que la première solution réponde parfaitement aux besoins de source routing, elle ne permet plus par la suite la mise en place du protocole de sécurité SERANO, qui réalise un chiffrement du chemin emprunté, voir la suppression des adresses IP, remplacées par des identifiants de nœud, afin de masquer aux nœuds traversés l'ensemble du chemin emprunté. La solution retenue est donc une modification de la pile TCP/IP afin d'ajouter une fonction de source routing.

Comme pour l'implémentation de la fonction de projection Mojette (cf. §7.1.1), la nouvelle fonction de routage est implémentée dans un module externe au noyau Linux. Ce dernier est modifié afin de contourner la fonction de routage classique si le nouveau module de routage est présent, permettant ainsi la validation des autres fonctions du protocole (Mojette, …) sans la présence de la fonction multi-chemins.

7.1.4 Implémentation du protocole SERANO

L'implémentation du protocole SERANO, comme spécifié dans le délivrable D22 (Spécification du protocole de sécurité SEREADMO : SERANO), est réalisé sous la pile TCP/IP. Les appels à la couche d'accès au réseau (Ethernet, 802.11, ...) du protocole IP sont donc détournés, sur le même principe que celui utilisé pour l'implémentation de la projection Mojette.

7.1.5 Remarques générales

7.1.5.1 Limites de l'expérimentation

Dans une première phase, étant données la complexité du protocole TCP (gestion des connexions, acquittement des trames, …) et le besoin de communiquer avec la machine d'analyse, seules les données du protocole UDP seront traitées par l'ensemble des modules décrits dans les paragraphes précédents. L'extension au protocole TCP sera réalisé dans une seconde phase, lorsque l'ensemble du protocole SEREADMO sera pleinement opérationnel et aura été validé sur les trames UDP.

7.1.5.2 Envoi des données d'analyse

Afin de permettre une analyse temps réel du comportement des nœuds du réseau, le système d'exploitation des ces derniers est modifié afin d'ajouter, en plus des fonctions spécifiques au protocole SEREADMO, des fonctions de mesure et de surveillance. Toutes les données ainsi collectées par ces fonctions seront envoyées par une application spécifique à la machine d'analyse.

7.2 Machine d'analyse

Comme indiqué au paragraphe 6.4 la machine d'analyse est du type PC serveur fonctionnant sous Linux. Et contrairement aux nœuds du réseau Ad Hoc aucune modification de son système d'exploitation n'est réalisée. Seul un outil de réception des données d'analyse en temps réel des nœuds du réseau est mis en œuvre.

Cet outil de collecte de données, développé pour les besoins du projet, permet :

- De recevoir les données d'analyse envoyées en temps réel par les nœuds,
- De stocker l'ensemble des données reçues pour une interprétation ultérieure
- D'afficher temps réel pour chacun des nœuds un résumé des données d'analyse
- De réaliser des contrôles de cohérence des données d'analyse.

Les données d'analyse envoyées par les nœuds sont les suivantes :

- Liste de routes connues par un nœud
- Données topologiques utilisées pour le calcul des routes (liste des nœuds voisins, voisins à 2 sauts,…)
- Nombre de paquets de données utilisateur envoyés et la liste leurs nœuds utilisés pour chaque envoi
- Nombre de paquets de données utilisateur reçus et retransmis ou supprimés.
- …

Cette liste des données n'est pas exhaustive est sera complété durant la phase d'expérimentation en fonction des besoins rencontrés. La communication réalisée entre les nœuds du réseau et le machine d'analyse étant dans un format générique (type XML) permettant un ajout simple et rapide de ces nouvelles données.

7.3 Le nœud espion

Ce nœud, dédié à l'interception des données du réseau Ad Hoc et à la génération d'un trafic de perturbation ne nécessite aucun développement particulier. Il utilise en effet, comme décrit au paragraphe 6.5 , le système d'exploitation Backtrack spécialement dédié à ce type de travail.

Il comportera, comme les nœuds du réseau, l'ensemble des modifications apportés au système afin d'implémenter le protocole SEREADMO pour pouvoir recevoir et émettre des données via ce protocole.

8. DEFINITION DES EXPERIMENTATIONS

Après la présentation de la plate-forme dans les chapitres précédents, ce chapitre détail maintenant les différentes expérimentations réalisées à l'aide de cette plate-forme, ainsi que la méthode employée pour la conduite de ces expérimentations.

8.1 Présentation de la méthode d'expérimentation

8.1.1 Introduction

La méthodologie utilisée pour la conduite de la phase d'expérimentation est issue de la méthodologie employée pour la validation de développements logiciels [KEOTEST]. Dans le cas du projet SEREADMO, cette méthodologie est adaptée afin de réaliser 2 types de validations :

- La validation de la plate-forme : les expérimentations réalisées durant cette phase doivent permettre de valider les développements réalisés. C'est à dire la validation du calcul des routes, le transfert des données utilisateur …
- La validation des simulations : les expérimentations réalisées durant cette phase doivent permettre de valider les capacités du protocole SEREADMO en situation réelle et de valider les résultats des simulations sous NS2.

8.1.2 Détail de la méthodologie

Cette méthodologie de test est utilisée pour valider la conformité de la conception du protocole et de sa mise en oeuvre par rapport aux objectifs initiaux du projet. Elle a pour but d'encadrer les différentes expérimentations réalisées afin de s'assurer que ces expérimentations couvriront l'ensemble des fonctions et capacités prévues dans la spécification du protocole SEREADMO et ceci dans les délais prévus par le projet.

Contrairement aux méthodes de test classiques, basées sur une approche par fonction du produit / protocole, la méthodologie de test utilisée est basée sur un découpage en domaines de préoccupation (sécurité, tenu en charge, …). Elle permet ainsi une approche progressive des tests et pallie aux problèmes de l'approche par fonctions lors de la validation de l'enchaînement de fonctions ou de la validation de fonctions composées.

Cette méthodologie se fonde sur la définition de types de test (ou domaines de préoccupation, ex : performances), décomposés en objectifs de test (ex : rendement par rapport au temps, rendement par rapport aux ressources), eux-mêmes décomposés en cas de test.

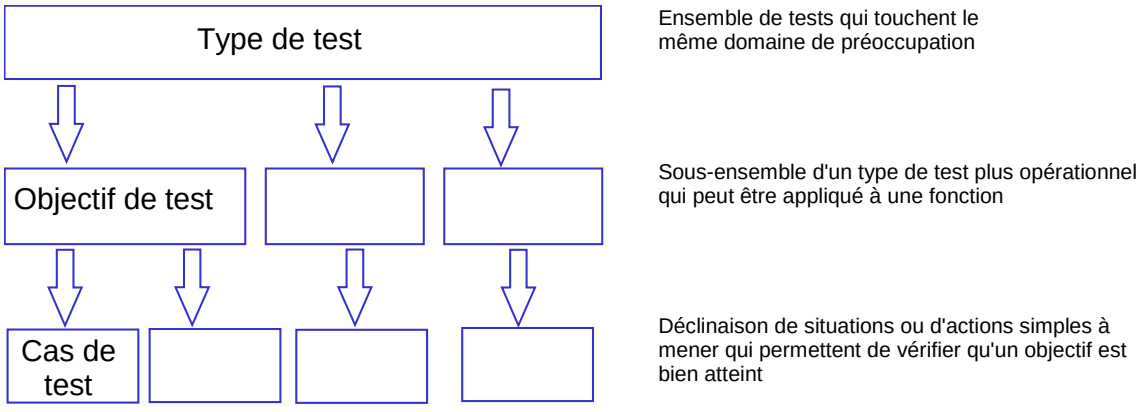

Figure 15 : Décomposition des tests de validation

Les cas de test ainsi définis sont réalisés à l'aide de scénarios. Ces scénarios définissent l'ensemble des actions basiques (cliquer sur un bouton, saisire une valeur, …) à effectuer de manière chronologique, ainsi que les résultats attendus, afin de réaliser les cas de test qu'ils représentent.

Le tableau ci-dessous donne un exemple de scénario pour la validation de la sécurité d'une application.

Actions à réaliser	Résultat attendu	Résultat obtenu
Lancer l'application	L'application se lance et propose une fenêtre d'identification	
Saisir le nom d'utilisateur 'YYY' et le mot de passe 'ZZZ' puis valider	Une fenêtre indique que le mot de passe n'est pas valide	
…	…	

Afin de s'assurer de la validation l'ensemble de fonctions d'une application, une matrice dite de couverture, regroupe en colonne l'ensemble des objectifs de test et en ligne l'ensemble des fonctions de l'application. La matrice ainsi définie permet de préciser la nécessité d'appliquer un objectif de test à une fonction donnée. La validation minimale de l'application étant réalisée lorsque chacune des inter-actions objectifs de test / fonctions ainsi identifiée est couverte par au moins un scénario.

La figure ci-dessous donne un exemple de matrice de couverture. Pour valider cette application, chacune des croix de la matrice symbolisant une interaction objectifs de test / fonction doit donc être couverte par au moins 1 scénario.

KEOSYS MEDICAL IMAGING — MATRICE DE COUVERTURE

Projet :	Types de tests ⇨	Interface					Dialogue H/M				
⇩ Fonctions	Objectifs de tests ⇨	I01	I02	I03	I04	I05	H01	H02	H03	H04	H05
	Identification utilisateur / Verrouillage session										
Gestion des dossiers	Affichage, filtre, tri et sélection dans la liste des examens	x					x	x			
	Ouverture de la sélection	x			x		x		x		
	Suppression de la sélection	x			x		x			x	
	Export de la sélection	x	x	x		x					
	Gravure de la sélection sur média CD / DVD	x					x				
import	Interrogation PACS (filtrage, critères)										
	Ouverture / Import PACS										
	Ouverture / Import CD/DVD DICOM										

Figure 16 : Exemple de matrice de couverture

Lorsque tous les scénarios ainsi définis ont été réalisés avec succès, la validation de l'application peut alors est considérée comme acceptable.

8.1.3 Objectifs de tests pour le protocole SEREADMO

Sur les bases de la méthodologie décrite au paragraphe précédent, les différents types de test et objectifs de test retenus pour la validation du protocole SEREADMO sont les suivants :

- Tests d'interface : validation de des données du protocole
  1. I01 : Validation de la présence des champs
  2. I02 : Validation de la conformité des champs

- Tests fonctionnels : validation de l'exécution correcte des fonctions en utilisation normale
  1. F01 : Validation de l'exécution correcte des fonctions
  2. F02 : Validation de l'exécution aux limites (ajout d'un premier nœud voisin, perte du dernier nœud voisin, …)
  3. F03 : Enchaînement (respect de règles de gestion, comportement du système lorsque l'utilisateur n'a pas le comportement attendu)
  4. F04 : Concurrence

- Tests de performance : analyse des performances
  1. PE1 : Rendement par rapport au temps : temps de réponse, …
  2. PE2 : Rendement par rapport aux ressources : espace mémoire occupé, …

- Test de charge : validation et analyse du comportement lors de l'utilisation aux limites

1. CH1 : Analyse de la baise de performance

- Tests des modes dégradés : Validation de l'aptitude à prévenir et à prendre en charge les dysfonctionnements de l'environnement.
    1. MD1 : Analyse de la robustesse et du fonctionnement en mode dégradé
    2. MD2 : Analyse des possibilités de récupération en cas d'arrêt transitoire ou d'arrêt prolongé

- Tests de sécurité :
    1. SE1 : Identification des utilisateurs
    2. SE2 : Résistance au percement

8.2 Application au protocole SEREADMO

Après avoir défini la méthodologie de test dans les paragraphes précédents, les prochains paragraphes vont maintenant l'appliquer à la validation du protocole SEREADMO. Pour cela, un premier paragraphe présente la matrice de couverture définie pour le projet et un second les différents scénarios obtenus à l'aide de cette matrice.

8.2.1 Matrice de couverture du projet

Dans le cadre de l'expérimentation, la matrices de couverture pour la validation du protocole est réalisée à l'aide des objectifs de test définis au paragraphe 8.1.3 et des fonctions suivantes du protocole :

- Détermination de la topologie du réseau
- Projection Mojette
- Routage multi-chemins
- Intégrité des données
- Authentification des nœuds
- Confidentialité des données

La matrice ainsi obtenue est la suivante :

MATRICE DE COUVERTURE													
Types de tests ⇨	Interface		Fonctionnels				Performances		Charge	Modes dégradés		Sécurité	
⇩ Fonctions / Objectifs de tests ⇨	I01	I02	FO1	FO2	FO3	FO4	PE1	PE2	CH1	MD1	MD2	SE1	SE2
Détermination de la topologie	x	x	x	x	x	x	x	x	x				
Projection Mojette	x	x	x	x			x	x	x	x			
Routage multi-chemins	x	x	x	x	x	x	x	x	x	x	x		
Intégrité de données	x	x	x		x								x
Authentification des nœuds	x	x	x		x							x	x
Confidentialité des données			x		x								x

Figure 17 : Matrice de couverture pour le protocole SEREADMO

8.2.2 Scénarios de test

Sur la base de la matrice de couverture définie précédemment, plusieurs scénarios sont définis et présentés ci-dessous. Cependant, l'objectif de ce délivrable n'étant pas de détailler ces scénarios (liste des actions basiques), seuls leurs intitulés et leurs prises en charge de la matrice de couverture sont présentés.

Afin de réaliser progressivement l'expérimentation, cette dernière est décomposée en plusieurs phases. Cette décomposition, permet de débuter la validation sur une version simplifiée de la plate-forme en nombre d'équipements et en fonctions, puis de progressivement la complexifié afin d'aboutir à la plate-forme complète prévu pour le projet. Ces différentes phases sont décrites ci-dessous avec les intitulés des différents scénarios mis en œuvres.

INTERIEUR 01 : Essais d'intégration sur une plate-forme simple composée de 2 PC en réseau LAN. La vérification porte sur la bonne transmission des paquets au travers des couches SEREADMO : Mojette, routage, sans la prise en compte du protocole de sécurité SERANO..

Scénario 1 : Découverte du réseau
- Détermination de la topologie : I01, I02, F01, F02, F03, F04

Scénario 2 : Transfert de données sans perturbation
- Détermination de la topologie : F01
- Projection Mojette : I01, I02, F01, PE1, PE2, CH1

INTERIEUR 02 : Essais d'intégration sur une plate-forme multipostes composée de 4 PC en réseau LAN et mise en œuvre de VLAN pour simuler le média de communication.

Scénario 1 et 2 : identique à la phase INTERIEUR 01

Scénario 3 : Mise en œuvre du multi-chemins sans perturbations
- Détermination de la topologie : F01
- Projection Mojette : F01, F02
- Routage multi-chemin : I01, I02, F01, F02, F03, F04

Scénario 4 : Mise en œuvre du multi-chemins avec perte de paquets, sans changement de topologie
- Détermination de la topologie : F01
- Projection Mojette : F01, MD1
- Routage multi-chemin : F01

Scénario 5 : Mise en œuvre du multi-chemins avec perte de paquets du au changement de topologie
- Détermination de la topologie : F01
- Projection Mojette : F01, F02, MD1
- Routage multi-chemin : F01, MD1, MD2

INTERIEUR 03 : Phase identique à la phase INTERIEUR 02 avec mise en œuvre du protocole de sécurité SERANO.

Scénario 1 à 5 : identique à la phase INTERIEUR 02

Scénario 6 : Transfert de données sans perturbation, avec interception des paquets par le nœud espion
- Intégrité des données : I01, I02, F01, F03
- Authentification des nœuds : I01, I02, F01, F03
- Confidentialité des données : F01, F03

Scénario 7 : Interception des paquets par le nœud espion et tentatives de déchiffrement
- Confidentialité des données : F01, SE2

Scénario 8 : Génération de paquets de perturbation par le nœud espion
- Intégrité des données : F01, SE2
- Authentification des nœuds : F01, SE1, SE2
- Confidentialité des données : SE2

INTERIEUR 04 : Expérimentation sur l'ensemble des postes prévus pour la plate-forme et recueil des métriques de référence.

Scénario 1 à 8 : identique à la phase INTERIEUR 03

Scénario 9 : Transfert de gros volumes de données sans perturbation
- Détermination de la topologie : F01, PE1, PE2, CH1
- Projection Mojette : F01, PE1, PE2, CH1
- Routage multi-chemin : F01, PE1, PE2, CH1

EXTERIEUR 05 : Essais d'intégration sur une plate-forme multipostes composée de 4 PC en champ libre, sans le protocole de sécurité SERANO.

Les scénarios sont identiques à la phase INTERIEUR 02

EXTERIEUR 06 : Expérimentation sur la plate-forme complète en champ libre, avec et sans le protocole de sécurité SERANO et recueil des métriques.

Les scénarios sont identiques à la phase INTERIEUR 04

EXTERIEUR 07 : Expérimentation sur la plate-forme complète en situation urbaine sur le site de la chantrerie à Nantes, avec et sans le protocole de sécurité SERANO et recueil des métriques.

Les scénarios sont identiques à la phase INTERIEUR 04

Les différents scénarios présentés si dessus sont définis afin de couvrir l'ensemble des points identifiés par la matrice de couverture, c'est à dire permettre une validation au moins minimal de protocole. D'autres scénarios pourront être ajoutés, lors des différentes phases d'expérimentation, si de nouveaux cas de test pertinents ou des besoins complémentaires sur les métriques sont identifiés.

9. REFERENCES

[6WIND] : An embedded networking software company, http://www.6wind.com/

[BACK] : Remote-Exploit.org - Supplying offensive security products to the world, http://www.remote-exploit.org/backtrack.html

[CAENSF] : Caen sans fil, http://www.caensansfil.org

[CRCOLSR] : CRC – Routing protocols for MANET, http://www.crc.ca/en/html/manetsensor/home/research_area/routing

[FRAWLS] : France Wireless, http://wireless-fr.org/

[HERBERT] Thomas F. Herbert , The Linux TCP/IP Stack: Networking for Embedded Systems

[INS2000] : Mark E. Russinovich, David A. Solomon. Inside Mircrosoft® Windows® 2000,

[KEOTEST] : Méthodologie des tests de validation, support de formation, Keosys

[LILLESF] : Lille sans file, http://www.lillesansfil.org

[LUCEOR] : Luceor, http://www.luceor.com/

[NANTESW] : Nantes Wireless, http://www.nantes-wireless.org/actu/

[NOAOLSR] : No Overhead Autoconfiguration OLSR, http://hipercom.inria.fr/noa-olsr/

[NROLSR] : The NRL OLSR Routing Protocol Implementation, http://cs.itd.nrl.navy.mil/work/olsr/

[OLSR] « RFC 3626 - Optimized Link State Routing Protocol (OLSR) », T. Clausen, P. Jacquet, INRIA, October 2003,

[OLSRD] : Olsrd, an adhoc wireless mesh routing daemon, http://www.olsr.org/

[OLSRNG] : Olsr-NG, http://wiki.funkfeuer.at/index.php/OLSR-NG

[OLSRV3] : OLSRv3, http://hipercom.inria.fr/olsr/

[OOLSR] : OOLSR, OLSR implementation, http://hipercom.inria.fr/OOLSR/

[OPENWRT] : OpenWrt Wirelsess Freedom, http://openwrt.org/

[PYOLSR] : OLSR in the Python language, http://hipercom.inria.fr/pyOLSR/

[QOLSR] : QOLSR, http://qolsr.lri.fr/

[RESCIT] : Réseau Citoyen, http://www.reseaucitoyen.be

[SMOLSR] : Simple Multicast OLSR, http://menetou.inria.fr/SMOLSR-MOLSR/

[TCPFOUND] : Andrew G. Blank, TCP/IP Fundations

[TCPILLUSTR] : W. Richard Stevens, TCP/IP Illustrated, Volume 1: The Protocols, Première édition

[TONNESEN] Andreas Tønnesen, Implementing and extending the optimized link state routing protocol.

[UPVA] : Polytechnic University of Valencia , OLSR protocol implementation, http://www.grc.upv.es/6software.html

[WININT] : Mark E. Russinovich, David A. Solomon. Microsoft® Windows® Internals, Fourth Edition: Microsoft Windows Server™ 2003, Windows XP, and Windows 2000