RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Abderrahmane Mira de Béjaïa

Mémoire de fin de cycle

Pour l'obtention du Diplôme de Master en Informatique
Option : Réseaux et Systèmes Distribués

Thème

Un Cadre Formel Pour la Prévention d'Intrusions

Présenté par :
AMOURI Salah
AKIK Yacine

Devant le jury composé de :

Président : M^{r} SAADI Mustapha
Examinateur 1 : M^{elle} AITABDLOUAHAB Karima
Examinateur 2 : M^{elle} YAHIAOUI Soraya
Encadreur : M^{elle} HAMZA Lamia

Année 2013

Remerciements

Nous remercions avant tout, Dieu tout-puissant qui nous a donné la force, le courage et la volonté pour réaliser ce travail.

Nous tenons à remercier notre promotrice M^{elle} HAMZA de nous avoir encadrés durant cette année, pour sa disponibilité, sa patience, son suivi et ces précieux conseils, et qui a su nous faire profiter de sa grande expérience.

Nous remercions tout particulièrement les membres de jury, qui ont accepté de juger notre travail.

Nous ne pourrons clôturer ces remerciements sans nous retourner vers les êtres qui nos sont les plus chers, qui ont eu un rôle essentiel et continu pendant notre réussite. Nous adressons de tout cœur nos remerciements à nos familles pour leur préoccupation et le souci qu'ils se sont fait pour nous, leur encouragement et leur suivi. Enfin, nous remercions tous ceux qui ont contribué de prés ou de loin à la réalisation de ce travail et particulièrement M^r HAROUNE Lamine.

Dedicaces

Je dédié ce modeste travail à ma mère et à la mémoire de mon père

A tous mes frères et sœurs

A toute ma famille en particulier Amine et Mustapha

A tous mes amis en particulier Mounir, Amine et Yahia

A mon binôme Salah avec qui j'ai partagé le meilleur et le pire.

AKIK Yacine.

Je dédié ce modeste travail à ceux que j'ai de plus chers : mes parents qui m'ont soutenu durant tout mon parcours d'études.

A mon frère.

A toute ma grande famille sans exception.

A tous mes amis en particulier Amine et Yahia

Et bien sûr, à mon binôme Yacine.

AMOURI Salah.

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

SSI	**S**écurité des **S**ystèmes d'**I**nformation
SI	**S**ystèmes d'**I**nformation
PSSI	**P**olitique de **S**écurité des **S**ystèmes **I**nformatiques
DOS	**D**enial **O**f **S**ervice
IP	**I**nternet **P**rorocol
DES	**D**ata **E**ncryption **S**tandard
IDEA	**I**nternational **D**ata **E**ncrypting **A**lgorithmus
TCP	**T**ransmission **C**ontrol **P**rotocol
HTTP	**H**yper **T**ext **T**ransfer **P**rotocol
FTP	**F**ile **T**ransfer **P**rotocol
IDS	**I**ntrusion **D**etection **S**ystem
N-IDS	**N**etwork based **I**ntrusion **D**etection **S**ystem
H-IDS	**H**ost based **I**ntrusion **D**etection **S**ystem
LAN	**L**ocal **A**rea **N**etwork
DMZ	**D**emilitarized **Z**one
VPN	**V**irtual **P**rivate **N**etwork
ATM	**A**synchronous **T**ransfer **M**ode
MPLS	**M**ulti **P**rotocol **L**abel **S**witching
PPTP	**P**oint to **P**oint **T**unneling **P**rotocol
L2F	**L**ayer **T**wo **F**orwarding
L2TP	**L**ayer **T**wo **T**unneling **P**rotocol
MANET	**M**obile **A**d hoc **NET**work
IETF	**I**nternet **E**ngineering **T**ask **F**orce
AODV	**A**d hoc **O**n-demand **D**istance **V**ector
OLSR	**O**ptimised **L**ink **S**tate **R**outing

ZRP	**Z**one **R**outing **P**rotocol
CBRP	**C**luster **B**ased **R**outing **P**rotocol
CCS	**C**alculus of **C**ommunicating **S**ystems
CSP	**C**ommunicating **S**equential **P**rocesses
ACP	**A**lgebra of **C**ommunicating **P**rocesses
BNF	**B**ackus-**N**aur **F**orm
SSA	**S**ecure **S**afe **A**mbient
RSC	**R**enforcement **S**ecurity **C**alculus
JDK	**J**ava **D**evelopement **K**it
JVM	**J**ava **V**irtuel **M**achine
EDI	**E**nvironnement de **D**éveloppement **I**ntégré

Introduction Générale

Les systèmes informatiques sont devenus aujourd'hui des outils indispensables pour le bon fonctionnement et l'évolution de la plupart des entreprises, et ils sont déployés dans différents domaines comme la banque, la médecine ou encore le domaine militaire. L'accroissement de l'interconnexion de ces derniers les a rendus accessibles par une population diversifiée d'utilisateurs qui ne cesse d'augmenter. Ces utilisateurs, connus ou non, ne sont pas forcément pleins de bonnes intentions vis-à-vis de ces systèmes. En effet, ils peuvent essayer d'accéder à des informations sensibles pour les lire, les modifier ou les détruire ou encore tout simplement pour porter atteinte au bon fonctionnement du mécanisme. Dès lors que ces systèmes sont apparus comme des cibles d'attaques potentielles, les sécuriser est devenu un enjeu incontournable.

Ce problème persiste d'avantage ce qui fait appel à la nécessité de développer des outils formels pour modéliser les réseaux informatiques sous la forme d'un langage mathématique. Ces outils nous permettraient une configuration sécurisée de ces réseaux. Des travaux plus récents tentent de sécuriser un système informatique ou/et un réseau informatique ; selon des méthodes formelles utilisant un calcul mathématique et des preuves de validation formelles. Parmi ces systèmes les réseaux AD HOC sont des réseaux sans fil capables de s'organiser sans infrastructure définie préalablement, ils sont caractérisés par leur sécurité limitée et leurs vulnérables aux diverses attaques.

L'objectif de ce mémoire est de modéliser l'une des attaques DoS dans les réseaux AD HOC (attaque du blackhole) et d'en déterminer une solution appropriée à cette attaque.

Organisation du mémoire

Ce mémoire est structuré de la manière suivante :

- Dans le chapitre 1, nous présentons des généralités sur la sécurité informatique ainsi qu'un petit aperçu sur la sécurité dans les réseaux AD HOC.
- Dans le chapitre 2, nous dressons un état de l'art sur l'algèbre de processus. Tout d'abord, nous présentons les quatre algèbres de processus (CCS, CSP, π-Calcul et le Calcul ambient), Par la suite nous montrons les travaux antérieurs utilisant le calcul ambient pour répondre aux différentes questions liées au contrôle d'accès.
- À travers le chapitre 3, nous montrons l'approche RSC, par la suite nous la discutons. Après, nous élaborons une spécification de l'attaque blackhole dans un réseau AD HOC, nous analysons le comportement d'un processus malicieux dans ce système et enfin nous terminons par une proposition pour améliorer la sécurité du réseau.
- Le chapitre 4 est consacré à la réalisation. Nous présentons l'application et les outils de développement utilisés pour implémenter les règles de réductions.
- Finalement, nous concluons par une récapitulation du travail accompli et quelques perspectives. Ainsi qu'un annexe, où nous allons modéliser deux autres systèmes informatiques.

Chapitre 1

Généralités sur la Sécurité Informatique

1.1 Introduction

L'évolution rapide des réseaux informatiques, privés ou publics, engendre un volume toujours plus important de données sauvegardées et transmises, générant ainsi de nouveaux besoins en matière de sécurité. Dans un monde où l'entreprise dépend de plus en plus de son système informatique, la sécurité est donc devenue une préoccupation primordiale. En effet, de nombreuses méthodes et outils ont été développés durant ces dernières années pour améliorer la sécurité informatique.

Dans ce chapitre nous étudierons les sections suivantes : La section 1.2, décrit les notions de base de la sécurité informatique. La section 1.3 présente les différentes menaces et leurs origines. La section 1.4 décrit la politique de sécurité informatique. La section 1.5 et la section 1.6 présentent les vulnérabilités et les diverses attaques. La section 1.7 traite l'architecture de la sécurité informatique. La section 1.8, désigne quelques techniques de défense et de sécurité. Dans la section 1.9 nous intéressons à la sécurité dans les réseaux AD HOC, et nous terminons par une conclusion.

1.2 La sécurité informatique

Aujourd'hui, les systèmes et les réseaux informatiques sont devenus des outils indispensables au fonctionnement et à l'évolution de toutes les activités des entreprises. Internet de sa part, relie des millions d'ordinateurs à travers le monde fonctionnant sur des plateformes multiples de matériel et de logiciel, avec des communications caractérisées par une offre multiservice[1]. Donc, il est important de souligné la nécessité

de la sécurité. Cette dernière ne devrait pas être une mission secrète ni reposer sur la peur et les représailles ; elle devrait être transparente, évolutive et motivante et avant tout naitre de la connaissance et de la compréhension.

La sécurité informatique à plusieurs objectifs liés aux types de menaces ainsi qu'aux types de ressources. Néanmoins, les principaux sont les suivants :

- Empêcher la divulgation non-autorisé de données.
- Interdire la modification non-autorisée de données.
- Empêcher l'utilisation non-autorisée de ressources réseau ou informatiques de façon générale.
- Etc.

1.2.1 Que faut-il sécuriser ?

Avant de réaliser un système de sécurité il faut d'abord se préoccuper de savoir ce qu'il faut protéger. On dénombre trois choses impotentes, à savoir :

1.2.1.1 Données

Ce sont les informations qu'on garde sur les ordinateurs, ces données possèdent trois caractéristiques qui justifient leur protection[1] :

- **Les secrets** : on ne désire probablement pas que d'autres personnes les connaissent.
- **La disponibilité** : la disponibilité de données permet de maintenir le bon fonctionnement du système d'information[2].
- **L'intégrité** : l'intégrité des données correspond à l'état existant lorsqu'une donnée est identique à sa source et qu'elle n'a pas été exposée à des altérations accidentelles ou malicieuses.

1.2.1.2 Ressources

Ce sont les ordinateurs eux-mêmes. La plupart des gens veulent utiliser leurs ordinateurs, ou veulent faire payer les autres pour leurs usages. Les ressources informatiques coûtent du temps et de l'argent, et chacun a le droit de déterminer par lui-même la façon d'utiliser les siennes. Ces ressources ne sont pas des ressources naturelles, mais

des ressources propres qui peuvent être gâchées ou détruites ou encore être utilisées à des fins malveillants[1].

1.2.1.3 Réputation

Un intrus peut apparaitre sur Internet avec une autre identité, et tout ce qu'il fait semble provenir du réseau alors qu'il n'en est rien.

1.2.2 Les objectifs de la sécurité informatique

La sécurité informatique est l'ensemble des moyens mise en œuvres pour réduire les vulnérabilités d'un système contre des menaces accidentelles ou intentionnelles, ainsi elle doit offrir une garantie contre les utilisateurs indélicats en les contrôlant selon des différentes fonctions[2]. Les principaux objectifs à garantir sont :

1.2.2.1 L'intégrité

L'intégrité permet d'assurer que l'information ne peut être modifiée que par les personnes autorisées.

1.2.2.2 La confidentialité

Permet de protéger le contenu des informations sauvegardées ou transmises sur un réseau, dont la diffusion doit être limitée à seules personnes autorisées.

1.2.2.3 L'authentification

Consiste à assurer l'identité d'un utilisateur, c'est-à-dire de garantir à chacun des correspondants que son partenaire est bien celui qu'il croit être. On distingue deux types d'authentification :

- **Authentification d'un tiers** : c'est l'action qui consiste à prouver son identité. Ce service est généralement rendu par l'utilisation d'un échange d'authentification qui implique un certain dialogue entre les tiers communicants.
- **Authentification de l'origine des données** : elle sert à prouver que les données reçues ont bien été émises par l'émetteur déclaré.

1.2.2.4 La non-répudiation

l'expéditeur ne peut nier avoir émis un message une fois le message est reçu par son destinataire[3].

1.3 Les menaces et leurs origines

La sécurité des systèmes d'information (SSI) est une discipline très importante car le système d'information (SI) est pour toute entreprise un élément absolument vital. Les menaces contre les systèmes d'information entrainent un risque de l'une des catégories suivantes : atteinte à la disponibilité des systèmes et des données, destruction des données, corruption ou falsification de données, vol ou espionnage de données, usage élicite d'un système ou d'un réseau, usage d'un système compromis pour attaquer d'autres cibles. Les menaces sont des adversaires déterminés, capable d'effectuer une attaque exploitant une vulnérabilité[4]. Elles sont produites de différentes origines :

1.3.1 Origines physiques

Elles peuvent être d'origines naturelle ou criminelle, comme :

- Désastre naturel (inondation, séisme, incendie).
- Panne matérielle.
- Panne de réseau.

1.3.2 Origines humaines

Elles peuvent être intentionnelles ou fortuites :

- Installation de softwares « troués ».
- Navigation web non maitrisée.
- Modification d'une configuration.

1.3.3 Origines opérationnelles

Elles sont liées à un état du système :

- Bug logiciel : Buffer Over flow.

– Disfonctionnement logiciel.

– Problèmes de configuration.

La plupart de ces problèmes sont traduis par des failles provoquant les disponibilités d'attaque telles que les dénis de services, qui laissent des portes ouvertes aux pirates, et cela est dû généralement à l'absence d'une politique de sécurité efficace.

1.4 La politique de sécurité informatique

Une politique de sécurité des systèmes informatiques (PSSI) doit déterminer les objets à sécuriser ; elle identifie les menaces à prendre en compte ; elle définit le périmètre de sécurité ; elle spécifie l'ensemble des lois, règlements et pratiques qui régissent la façon de gérer, protéger et diffuser les informations et autres ressources sensibles au sein d'un système spécifique, d'une entité. Ainsi la PSSI permet d'avoir une approche méthodique et systématique pour garantir une sécurité homogène de son système informatique[5].

1.4.1 La politique de sécurité réseaux

Une politique de sécurité réseau est un document générique qui : définit des règles à suivre pour les accès aux réseaux informatiques et aux flux autorisés ou non, détermine comment les politiques sont appliquées, et présente une partie de l'architecture de base de l'environnement de sécurité du réseau. La politique de sécurité réseau est un des éléments de la politique de sécurité informatique, qui est elle-même un des éléments de la politique de sécurité du système d'information [5]. C'est la raison pour laquelle il est nécessaire de fixer dans un premier temps un périmètre de sécurité et élaborer une politique de sécurité.

1.4.2 Fixer un périmètre de sécurité

Inutile de se préoccuper de sécurité sans avoir défini ce qui est à protéger : en d'autre termes toutes organisation désireuse protéger ses systèmes et ses réseaux doit déterminer son périmètre de sécurité. Le périmètre de sécurité, au sein de l'univers physique, délimite l'intérieur et l'extérieur, mais sa définition doit aussi englober (ou pas)

les entités immatérielles qui peuplent les ordinateurs et les réseaux, essentiellement les logiciels et en particulier les systèmes d'exploitation. Une fois fixé ce périmètre, il faut aussi élaborer une politique de sécurité, c'est-à-dire décider de ce qui est autorisé et de ce qui est interdit. À cette politique viennent en principe s'ajouter les lois et les règlements en vigueur, qui s'imposent à tous.

Si avec l'aide du service juridique de votre entreprise vous avez réussi à surmonter ces difficultés et à mettre sur pieds une politique de sécurité des systèmes d'information, il vous sera possible de mettre en place les solutions techniques appropriées à la défense du périmètre selon la politique choisie. Mais déjà il est évident que les dispositifs techniques ne pourront pas résoudre tous les problèmes de sécurité, et, de surcroit, la notion même de périmètre de sécurité est aujourd'hui battu en brèche par des phénomènes comme la multiplication des ordinateurs portables et autres objets mobiles informatiques en réseau qui, par définition, se déplace de l'intérieur à l'extérieur et inversement , à quoi s'ajoute l'extraterritorialité de fait des activités sur Internet[6].

1.4.3 But de la politique de sécurité

La politique de sécurité informatique fixe les principes visant à garantir la protection des ressources informatiques et de télécommunications en tenant compte des intérêts de l'organisation et de la protection des utilisateurs.
Les ressources informatiques et de télécommunications doivent être protégées afin de garantir la confidentialité, l'intégrité et la disponibilité des informations qu'elles traitent, dans le respect de la législation contre toutes les menaces et les attaques dont les vulnérabilités sont exploitées[7].

1.5 Les vulnérabilités

Une vulnérabilité est une faille dans un système informatique, permettant à un attaquant de porter atteinte à l'intégrité du système.

1.6 Les attaques

Tout ordinateur connecté à un réseau informatique est potentiellement vulnérable à une attaque informatique.

1.6.1 Les attaques informatiques

Une attaque informatique est l'exploitation d'une faille d'un système informatique (système d'exploitation, logiciel...etc.) à des fins non connues par l'exploitant des systèmes et généralement préjudiciables. Les motivations des attaques peuvent être de différentes sortes [8] :

- obtenir un accès au système.
- voler des informations, tels que des secrets industriels ou des propriétés intellectuelles.
- glané des informations personnelles sur un utilisateur.
- récupérer des données bancaires.
- troubler le bon fonctionnement d'un service.
- utiliser le système de l'utilisateur comme "rebond" pour une attaque.
- utiliser les ressources du système de l'utilisateur, notamment lorsque le réseau sur lequel il est situé possède une bande passante élevée.

1.6.2 Les types des attaques réseaux

Il existe plusieurs techniques d'attaques qui peuvent être regroupées en trois familles différentes :

1.6.2.1 Les attaques directes

C'est les plus simples des attaques dans le monde des pirates informatique, le pirate attaque directement sa victime à partir de son ordinateur. Par ce type d'attaque, il y a de grandes chances pour que la victime puisse remonter à l'origine de l'attaque pour identifier l'identité de l'attaquant[9].

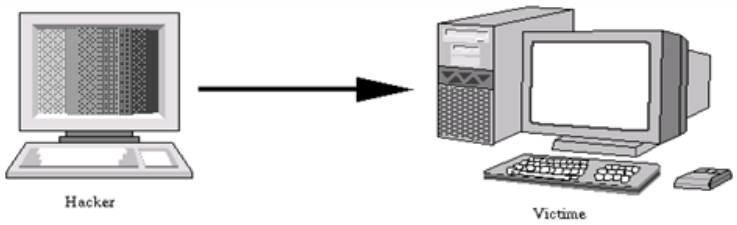

FIGURE 1.1: Attaque directe

1.6.2.2 Les attaques indirectes par rebond

Le principe de cette technique est d'envoyer des paquets à un ordinateur intermédiaire dans le but de masquer l'identité du hacker et aussi d'utiliser ses ressources puisque il est plus puissant. Pour attaquer, cet ordinateur intermédiaire répercute l'attaque sur la victime d'où le terme rebond. L'avantage de cette technique réside dans la difficulté de remonter jusqu'à l'attaquant, puisque dans le meilleur des cas vous remontrez à l'ordinateur intermédiaire[9].

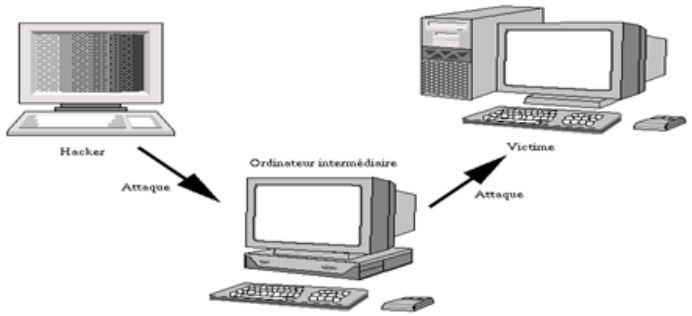

FIGURE 1.2: Attaque indirecte par rebond

1.6.2.3 Les attaques indirectes par réponse

Cette attaque est une dérivée de l'attaque par rebond. Elle offre les mêmes avantages, du point de vue du hacker. Mais au lieu d'envoyer une attaque à l'ordinateur intermédiaire pour qu'il la répercute, l'attaquant va lui envoyer une requête. Et c'est cette réponse à la requête qui va être envoyée à la machine victime[9].

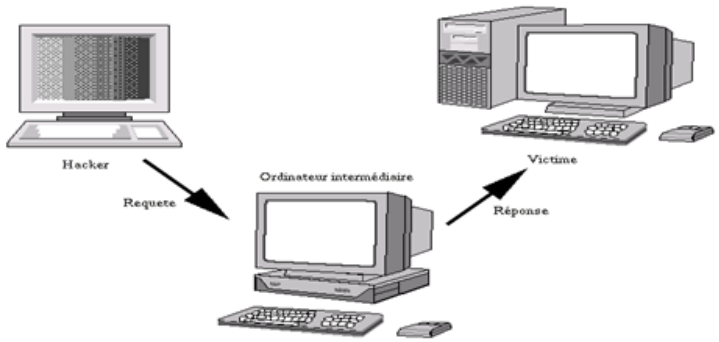

FIGURE 1.3: Attaque indirecte par réponse

1.6.3 Quelques exemples d'attaques

Il existe plusieurs types d'attaques réseaux ayant de mauvaises conséquences telles que la mise hors-service d'un système et l'interception des données. Nous citons dans ce qui suit quelques attaques :

- **Le sniffing** : Cette attaque est utilisée par les pirates informatiques pour obtenir des mots de passe. Grâce à un logiciel appelé (sniffer), cette technologie n'est pas forcement illégale car elle permet aussi de détecter des failles sur un système, mais elle devient une arme pour le hacker. Elle est cependant limitée, car ce dernier doit se trouver sur le même réseau de la machine qu'il veut pirater[10].
- **Les attaques par saturation (Déni de service DOS)** : Les attaques par saturation sont des attaques informatiques qui consiste à envoyer des milliers de messages depuis des dizaines d'ordinateurs, dans le but de submerger les serveurs d'une société, de paralyser pendant plusieurs heures son site Web et d'en bloquer ainsi l'accès aux Internautes. Il existe des différentes attaques par saturation :
  - ***Le flooding*** : Cette attaque consiste à envoyer à une machine de nombreux paquets IP de grosse taille.
  - ***Le TCP-SYN flooding*** : l'envoie d'un grand nombre de demande de connexions au serveur (SYN) à partir de plusieurs machines.
  - ***Le smurf*** : s'appuie sur le ping et les serveurs de broadcast. On falsifie d'abord son adresse IP pour se faire passer pour la machine cible. On envoie alors un ping sur un serveur de broadcast. Il le fera suivre à toutes les machines qui

sont connectées qui renverront chacune un « pong » au serveur qui fera suivre à la machine cible.

- **Le débordement de tampon** : l'envoie à la machine cible des données d'une taille supérieure à la capacité d'un paquet[1].

- **Le IP spoofing** :

  L'usurpation d'adresse IP (en anglais : IP spoofing ou IP address spoofing) est une technique utilisée en informatique qui consiste à envoyer des paquets IP en utilisant une adresse IP source qui n'a pas été attribuée à l'ordinateur qui les émet. Le but peut être de masquer sa propre identité lors d'une attaque d'un serveur, ou d'usurper en quelque sorte l'identité d'un autre équipement du réseau pour bénéficier des services auxquels il a accès[11].

- **Le craquage de mot de passe** : consiste à faire de nombreux essais jusqu'à trouver le bon mot de passe[10]. Il existe deux grandes méthodes :
  - L'utilisation de dictionnaires : le mot testé est pris dans une liste prédéfinie contenant les mots de passe les plus courants et aussi des variantes de ceux-ci (à l'envers, avec un chiffre à la fin...). Les dictionnaires actuels contiennent dans les 50 000 mots et sont capables de faire une grande partie des variantes.
  - La méthode brute : toutes les possibilités sont faites dans l'ordre jusqu'à trouver la bonne solution.

- **Les chevaux de Troie(Trojans)** : Les chevaux de troie sont des programmes informatiques cachés dans d'autres programmes. En général, le but d'un cheval de Troie est de créer une porte dérobée (backdoor) pour qu'un pirate informatique puisse ensuite accéder facilement à l'ordinateur ou au réseau informatique. Il peut aussi voler des mots de passe, copier des données, exécuter des actions nuisibles[10].

- **L'ingénierie sociale** : (social engineering en anglais) n'est pas vraiment une attaque informatique, c'est plutôt une méthode pour obtenir des informations sur un système ou des mots de passe. Elle consiste surtout à se faire passer pour quelqu'un que l'on est pas (en général un des administrateurs du serveur que l'on veut pirater) et de demander des informations personnelles (login, mots de passe, accès, numéros, données...)[12].

1.7 Architecture de sécurité informatique

On retrouve actuellement trop souvent des architectures de sécurité axées uniquement sur la prévention et la défense de périmètre. Il y a bien d'autres éléments qui doivent composer une architecture de sécurité informatique[13]. Toute architecture de sécurité (et plus globalement l'approche même de la sécurité) doit reposer sur un triptyque tel que :

- Prévention
- Détection
- Réaction

Ces trois aspects sont pour le moment très diversement couverts par le marché malgré une nécessité indéniable.

1.7.1 Prévention

La prévention est fondamentale et est généralement bien appréhendée par le plus grand nombre. Le principe : faire tout ce qu'il faut pour se protéger. Elle consiste le plus souvent à adopter la démarche suivante :

- Analyse des risques.
- Définition d'une politique de sécurité.
- Mise en œuvre d'une solution centrée sur un ou plusieurs firewalls.
- Audition de la solution.
- Mises à jour.

Le marché à ce jour couvre très bien cette approche : les cabinets de conseils sont très présents sur l'analyse des risques. Les Intégrateurs proposent et mettent place des solutions à tour de bras. Des sociétés se spécialisent dans la réalisation d'audits de sécurité, d'autres effectuent de la veille technologique en sécurité et permettent de déclencher les mises à jour (généralement effectuées par l'intégrateur).

1.7.2 Détection

Le principe est d'être capable de détecter lorsque les mesures de prévention sont prises par défaut. La détection, même si certains outils techniques existent, est encore

trop rarement intégrée aux infrastructures. Il est vrai que les intégrateurs proposent souvent ces outils lors de la mise en place d'infrastructures de connexion Internet ; mais leur déploiement reste marginal en dehors de ces projets spécifiques. De plus, à l'heure actuelle un cruel défaut de compétence est à déplorer. Il y à encore trop peu de personnes formées à ce type d'outils. La détection exige un suivi permanent de l'état des systèmes à protéger et des mécanismes de diffusion des alertes générées.

1.7.3 Réaction

S'il est important de savoir qu'une attaque est en cours ou qu'une attaque a réussi il est encore plus important de se donner les moyens de réagir à cet état de fait. C'est l'aspect le plus négligé actuellement même au sein des acteurs majeurs de la sécurité informatique. Pourtant, il n'est pas possible d'oublier les credo de tous les consultants en analyse de risque : " le risque zéro n'existe pas " ou encore " il n'y a pas de sécurité absolue ". Il faudrait donc toujours prévoir et se préparer au pire. Cela implique la mise en œuvre de procédures d'exploitation spécifiques à la réaction en cas d'attaque, la rédaction et le test d'un plan de continuité informatique à utiliser en cas de sinistre grave. Il est également primordial de se doter des outils permettant d'une part de collecter toutes les informations pouvant être nécessaires en cas de recours juridique. Un cadre doit aussi être prévu au niveau des responsabilités ; de ce fait les contrats d'assurance devront prendre en compte le risque représenté par les pirates. Le marché couvre très mal cet aspect à l'heure actuelle. Il n'existe que très peu de sociétés proposant une offre réelle en investigation d'incidents. Par ailleurs, même si certains cabinets de juristes se spécialisent dans le droit de l'Internet, la couverture du risque informatique et la définition des " éléments de preuve " dans les affaires de crimes informatiques restent encore floues.

1.8 Principales techniques de sécurité

De nos jours différentes techniques et méthodes ont été développées pour mettre en œuvre une stratégie de sécurité :

– Authentification.

- Cryptographie.
- Firewalls.
- Utilisation d'un proxy.
- Les anti-virus.
- Les systèmes de détection d'intrusions (IDS).
- La zone démilitarisée (DMZ).
- Les réseaux privés virtuels (VPN).

1.8.1 Authentification

On peut ranger les mécanismes d'authentification en trois catégories qui vérifient au moins l'un des critères :

- **Quelque chose que l'on est** : Comprend des techniques comme la prise d'empreintes digitales, l'analyse de la voix, la forme du visage, etc.
- **Quelque chose que l'on sait** : C'est le système de mot de passe traditionnel.
- **Quelque chose que l'on a** : Cela comprend des mécanismes comme des listes de questions-réponses, des cartes à puces, etc.

Certains systèmes combinent ces approches. En théorie il est bon de combiner au moins deux mécanismes, car il est rare d'avoir la possibilité de voler les deux choses à la fois[14].

1.8.2 Cryptographie

Dans cette technique de sécurité on distingue deux types :

1.8.2.1 Le chiffrement

Le chiffrement permet d'assurer la confidentialité des données en utilisant un système de clé appliqué sur les messages envoyés. Ces derniers sont décryptables par une clé unique correspondant au cryptage. Il existe deux grandes techniques de chiffrement :

1. **Le chiffrement symétrique (à clé secrète)** : cette technique est basée sur l'utilisation d'une clé partagée entre l'émetteur et le destinataire, cette clé sert à chiffrer et déchiffrer les messages. Plusieurs algorithmes de chiffrement utilisent

cette technique, les plus connus sont : Kerberos, DES (Data Encryption Standard) et IDEA (International Data Encrypting Algorithmus).

2. **Le chiffrement asymétrique (à clé publique)** : celle-ci utilise deux clés différentes : l'une des deux n'est connue que par l'utilisateur donc elle est privée, l'autre est publique et donc accessible par tout le monde. Les clés publiques et privées sont mathématiquement liées par l'algorithme de cryptage de telle manière qu'un message crypté avec une clé publique ne puisse être décrypté qu'avec la clé privée correspondante. Une clé est donc utilisée pour le cryptage et l'autre pour le décryptage[15].

1.8.2.2 Signature

Pour l'identification de l'émetteur du message on utilise la signature numérique et le certificat.

1. **Signature numérique :** Le principe de la signature numérique consiste à appliquer une fonction mathématique sur une portion du message. Cette fonction mathématique s'appelle fonction de hachage et le résultat de cette fonction est appelé code de hachage. Ce code fait usage d'empreinte digitale du message. Il est à noter que la fonction est choisie de telle manière qu'il sera impossible de changer le contenu du message sans altérer le code de hachage. Ce code de hachage est ensuite crypté avec la clé privée de l'émetteur et rajouté au message. Lorsque le destinataire reçoit le message, il décrypte ce code grâce à la clé publique de l'expéditeur puis il compare ce code à un autre code qu'il calcule grâce au message reçu. Si les deux correspondent, le destinataire sait que le message n'a pas été altéré et que son intégrité n'a pas été compromise. Le destinataire sait aussi que le message provient de l'émetteur puisque seul ce dernier possède la clé privée qui a crypté le code[14].

2. **Les certificats :**

   Un certificat est une structure de données qui est numériquement signée par une autorité certifiée en qui les utilisateurs peuvent faire confiance. Il contient une série de valeurs, comme le nom du certificat et son utilisation, des informations

identifiant le propriétaire et la clé publique, la clé publique elle-même, la date d'expiration et le nom de l'organisme de certificats. L'autorité certifiée utilise sa clé privée pour signer le certificat et assure ainsi une sécurité supplémentaire [16]. Si le récepteur connaît la clé publique d'une autorité certifiée, il peut vérifier que le certificat provient vraiment de cette autorité et il est assuré que le certificat contient donc des informations viables et une clé publique valide.

1.8.3 Firewall (Pare-feu)

Un firewall est l'un des dispositifs de protection, son principe est de construire un filtre entre un réseau local et un autre réseau non sûr tel que l'Internet ou un autre réseau local. Il constitue un point unique où l'audit et la sécurité peuvent être imposés. Tous les échanges passeront par lui. Il pourra donner des résumés du trafic, des statistiques sur ce trafic ou encore toutes les connexions entre les deux réseaux.
Deux objectifs sont visés par les firewalls : Le premier est de contrôler et protéger les hôtes du réseau local contre la divulgation non autorisée d'informations sensibles et contre les virus et les chevaux de Troie. Le deuxième objectif consiste à protéger les serveurs Internet contre des commandes jugées dangereuses associées à des services du type "telnet " et "sendmail ". Le firewall protège aussi le serveur contre la modification ou la suppression non autorisée de fichiers essentiels pour le système[17].

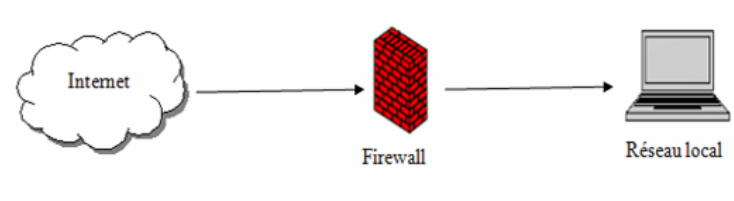

FIGURE 1.4: Firewall

Les firewalls possèdent plusieurs inconvénients :

- Un firewall ne protège pas des attaques qui ne passent pas par lui.
- Un Firewall n'est pas utile s'il existe dans le réseau des machines qui sont directement connectés par modem au monde extérieur (Internet).

– Un firewall ne peut pas protéger contre les attaques qui viennent de l'intérieur de l'entreprise.

1.8.4 Proxy

Un serveur proxy (traduction française de proxy server, appelé aussi serveur mandataire) est à l'origine une machine faisant fonction d'intermédiaire entre les ordinateurs d'un réseau local (utilisant parfois des protocoles autres que le protocole TCP/IP) et internet. La plupart du temps le serveur proxy est utilisé pour le web, il s'agit alors d'un proxy HTTP. Toutefois il peut exister des serveurs proxy pour chaque protocole applicatif (FTP, ...).

Le principe de fonctionnement d'un serveur proxy est assez simple : il s'agit d'un serveur "mandaté" par une application pour effectuer une requête sur Internet à sa place. Ainsi, lorsqu'un utilisateur se connecte à internet à l'aide d'une application cliente configurée pour utiliser un serveur proxy, celle-ci va se connecter en premier lieu au serveur proxy et lui donner sa requête. Le serveur proxy va alors se connecter au serveur que l'application cliente cherche à joindre et lui transmettre la requête. Le serveur va ensuite donner sa réponse au proxy, qui va à son tour la transmettre à l'application cliente[18].

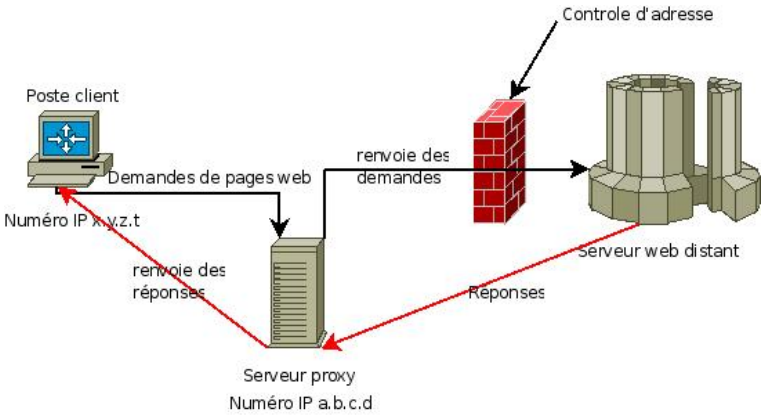

FIGURE 1.5: Proxy

1.8.5 Les antivirus

Les antivirus sont des logiciels conçus pour identifier, neutraliser et éliminer des logiciels malveillants (dont les virus ne sont qu'un exemple). Ceux-ci peuvent se baser sur l'exploitation des failles de sécurité, mais il peut également s'agir de programmes modifiant ou supprimant des fichiers, que ce soit des documents de l'utilisateur de l'ordinateur infecté, ou des fichiers nécessaires au bon fonctionnement de l'ordinateur. Les antivirus utilisent trois techniques pour détecter les virus :

1.8.5.1 Le Scanning

C'est la méthode la plus ancienne et la plus utilisée pour la recherche de virus. Elle consiste à rechercher un code spécifique qui indique la présence d'un virus. L'avantage de cette technique est sa capacité de détecter les virus avant même qu'ils s'exécutent sur l'ordinateur, mais elle nécessite une mise à jour à chaque fois que de nouveaux virus apparaissent.

1.8.5.2 Le Moniteur de comportement

Un moniteur de comportement est un programme résidant que l'utilisateur charge à partir du fichier AUTOEXEC.BAT et qui reste alors actif en arrière plan, surveillant tout comportement inhabituel[19].

1.8.5.3 Le contrôleur d'intégrité

Les contrôleurs d'intégrités signalent toute modification intervenue dans un fichier. Schématiquement un contrôleur d'intégrité construit un fichier contenant les noms de tous les fichiers présents sur le disque auxquels sont associées quelques caractéristiques (par exemple : la taille, la date et l'heure de la dernière modification).

1.8.6 Les systèmes de détection d'intrusions (IDS)

Un système de détection d'intrusion (IDS) est un logiciel qui réalise ses tâches en temps réel et est capable de détecter l'intrusion par combinaison de différentes techniques. Les IDS n'incluent pas la prévention contre les intrusions, il se contente de

les détecter et de les reporter à un opérateur ou de mener une action définit d'avance après qu'un intrus soit détecté ou de lancer une alarme[20].

Il existe deux grandes familles distinctes de système de détection d'intrusion :

- **Les N-IDS** *(Netwrk Based Intusion Detection System)* : ils assurent la sécurité au niveau du réseau.
- **Les H-IDS** *(Host Based Intusion Detection System)* : ils assurent la sécurité au niveau des hôtes.

1.8.7 La zone démilitarisée (DMZ)

Une zone démilitarisée (ou DMZ, en anglais Demilitarized Zone) est un sous-réseau isolé par un pare-feu. Ce sous-réseau contient des machines se situant entre un réseau interne (LAN) et un réseau externe (Internet). La DMZ permet à ses machines d'accéder à Internet et/ou de publier des services sur Internet sous le contrôle du pare-feu externe. En cas de compromission d'une machine de la DMZ, l'accès vers le réseau local est encore contrôlé par le pare-feu interne[17]. La figure suivante représente un cas particulier de DMZ où les deux pare-feu sont fusionnés :

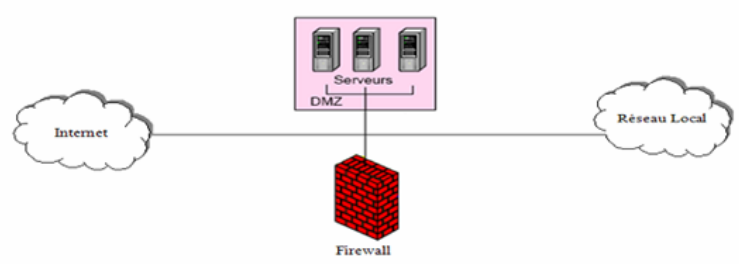

FIGURE 1.6: Architecture DMZ

La politique de sécurité mise en œuvre sur la DMZ est généralement la suivante :

- Trafic du réseau externe vers la DMZ autorisé.
- Trafic du réseau externe vers le réseau interne interdit.
- Trafic du réseau interne vers la DMZ autorisé.
- Trafic du réseau interne vers le réseau externe autorisé.
- Trafic de la DMZ vers le réseau interne interdit.
- Trafic de la DMZ vers le réseau externe refusé.

La DMZ possède donc un niveau de sécurité intermédiaire, mais son niveau de sécurisation n'est pas suffisant pour y stocker des données critiques pour l'entreprise. Il est à noter qu'il est possible de mettre en place des DMZ en interne afin de cloisonner le réseau interne selon différents niveaux de protection et ainsi éviter les intrusions venant de l'intérieur.

1.8.8 Les réseaux privés virtuels (VPN)

Dans les réseaux informatiques, le réseau privé virtuel (Virtual Private Network en anglais, abrégé en VPN) est vu comme une extension des réseaux locaux et préserve la sécurité logique que l'on peut avoir à l'intérieur d'un réseau local[20]. Il correspond en fait à une interconnexion de réseaux locaux via une technique de « tunnel ». On parle de VPN lorsqu'un organisme interconnecte ses sites via une infrastructure partagée avec d'autres organismes. Il existe deux types de telles infrastructures partagées : les « publiques » comme Internet et les infrastructures dédiées que mettent en place les opérateurs pour offrir des services de VPN aux entreprises. C'est sur Internet et les infrastructures IP que se sont développées les techniques de « tunnel ». Historiquement les VPN inter-sites sont apparus avec X.25 sur des infrastructures mises en place par les opérateurs, puis X.25 a été remplacé par le relayage de trames, l'ATM et le MPLS aujourd'hui.

Le principe de VPN est basé sur la technique de tunnelling. Cela consiste à construire un chemin virtuel après avoir identifier l'émetteur et le destinataire. Par la suite, la source chiffre les données et les achemine en empruntant ce chemin virtuel. Afin d'assurer un accès aisé et peu coûteux aux intranets ou aux extranets d'entreprise, les réseaux privés virtuels d'accès simulent un réseau privé, alors qu'ils utilisent en réalité une infrastructure d'accès partagée, comme Internet. Les données à transmettre peuvent être prises en charge par un protocole différent d'IP. Dans ce cas, le protocole de tunneling encapsule les données en ajoutant un en-tête. Le tunneling est l'ensemble des processus d'encapsulation, de transmission et de dés-encapsulation.
Il existe trois principaux protocoles de VPN :

1. PPTP (Point to Point Tunnelling Protocol) de Microsoft :

C'est un protocole de niveau 2 qui encapsule des trames PPP dans des datagrammes IP afin de les transférer sur un réseau IP. PPTP permet le cryptage des données PPP encapsulées mais aussi leur compression.

2. L2F (Layer Two Forwarding) de Cisc :
   L2F est un protocole de niveau 2 qui permet à un serveur d'accès distant de véhiculer le trafic sur PPP et de transférer ces données jusqu'à un serveur L2F (routeur). Ce serveur L2F désenchaîne les paquets et les envoie sur le réseau.
3. L2TP (Layer Two Tunnelling Protocol) de l'IETF :
   Ce protocole réunit les avantages de PPTP et L2F, il est souvent utilisé pour créer des VPN sur Internet. L2TP est un protocole réseau qui encapsule des trames PPP pour les envoyer sur des réseaux IP, X25, relais de trames ou ATM (Asynchronous Transfer Mode). Lorsqu'il est configuré pour transporter les données sur IP, L2TP peut être utilisé pour faire du Tunnelling sur Internet.
4. IP SEC :
   Est un protocole de niveau 3, permettant de transporter des données chiffrées pour les réseaux IP.

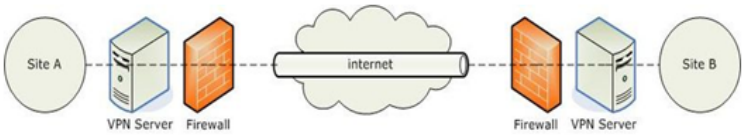

FIGURE 1.7: Principe de VPN

1.9 Sécurité dans les réseaux AD HOC

1.9.1 Définition d'un réseau AD HOC

Les réseaux AD HOC ont été largement étudiés par le groupe de travail MANET (Mobile Ad hoc NETwork) de l'IETF (Internet Engineering Task Force).
Ce dernier a défini formellement un réseau AD HOC comme étant un réseau formé dynamiquement par un ensemble de nœuds mobiles, qui sont reliés par l'intermédiaire

des liens sans fil, sans recours à une infrastructure préexistante ou à une administration centralisée. Les nœuds sont libres de se déplacer aléatoirement, et de s'organiser automatiquement, ainsi la topologie du réseau peut changer rapidement, d'une manière arbitraire et imprévisible[21].

1.9.2 Le routage dans les réseaux AD HOC

Le routage est une méthode d'acheminement des informations vers la bonne destination à travers un réseau de connexion donné. Il consiste à assurer une stratégie qui garantit, à n'importe quel moment, un établissement de routes qui soient correctes et optimales entre n'importe quelle paire de nœud appartenant au réseau. Ce qui assure l'échange des messages d'une manière continue. Le routage constitue un sérieux problème à résoudre pour que les réseaux sans fil puissent fonctionner dans de bonnes conditions[22].

1.9.2.1 Classification des protocoles de routage

Le routage dans les réseaux AD HOC est une opération complexe, cela est due éventuellement au changement fréquent de la topologie. De nombreux protocoles et algorithmes ont été proposés. Le groupe MANET, créé en 1997 par l'IETF(Internet Engineering Task Force), fait une distinction entre trois classes de protocoles : réactifs, proactifs et hybrides. Nous allons donner une brève explication de ces trois classes.

1.9.2.1.a Protocoles réactifs

Les protocoles de routage appartenant à cette classe, créent et maintiennent les routes selon les besoins. Lorsque le réseau a besoin d'une route, une procédure de découverte globale de route est lancée, et cela dans le but d'obtenir une information spécifiée, inconnue au préalable[22]. Le protocole réactif le plus connu est AODV(Ad hoc On-demand Distance Vector) standarisé en 2003 par l'IETF.

1.9.2.1.b Protocoles proactifs

Les protocoles de routage proactifs se basent sur la même philosophie des protocoles de routage utilisés dans les réseaux filaires classiques. Les deux principales méthodes

utilisées sont :

- La méthode de Lien (*Link State*).
- La méthode du Vecteur de Distance (*Distance Vector*).

Les deux méthodes exigent une mise à jour périodique des données de routage qui doit être diffusée par les différents nœuds du réseau. L'un des principaux protocoles de cette classe est OLSR (Optimised Link State Routing) qui a été standarisé par l'IETF (Internet Engineering Task Force) en avril 2004[22].

1.9.2.1.c Protocoles hybrides

Dans ce type de protocoles, on peut garder la connaissance locale de la topologie jusqu'à un nombre prédéfini (a priori petit) de sauts par un échange périodique de paquets de contrôle, autrement dit par une technique proactive. Les routes vers des nœuds plus lointains sont obtenues par un schémas réactif. Les protocoles ZRP (Zone Routing Protocol) et CBRP (Cluster Based Routing Protocol) font partie de cette classe[22].

1.9.3 Vulnérabilités des réseaux AD HOC

Les réseaux AD HOC sont plus vulnérables que les réseaux filaires à cause de ses propriétés qui se résument dans le fait que :

1. *Manque d'infrastructure* : Étant donné qu'il n'y a pas d'infrastructure, les réseaux AD HOC ne peuvent pas utiliser les équipements dédiés à la sécurité dans les réseaux traditionnels tels que les pare-feux ou les serveurs d'authentification. Tous les services de sécurité doivent donc être distribués et coopératifs[23].
2. *Bande passante limitée* : À cause des limitations de la bande passante, les communications peuvent facilement être perturbées, l'intrus peut effectuer cette attaque en occupant le support avec ses propres messages, ou tout simplement en perturbant les communications avec du bruit.
3. *Lien sans fil* : Quiconque possédant le récepteur adéquat peut potentiellement écouter ou perturber les messages échangés, à la différence dans les réseaux filaire où un intrus doit gagner l'accès physique au câble.

4. *Équivalence des nœuds de réseau* : Comme tous les nœuds du réseau AD HOC participent au routage, donc un nœud malicieux peut modifier, ajouter, supprimer les messages en transit, ce qui entraîne une perturbation du réseau.
5. *Contrainte d'énergie* : La consommation d'énergie constitue un problème important pour des équipements fonctionnant avec une alimentation autonome. Cette dernière vulnérabilité fait que les attaques par dénie de service (DoS), sont possibles.

1.9.3.1 Attaques contre les réseaux AD HOC au niveau du routage

Contrairement aux réseaux filaires, les réseaux AD HOC offrent une grande possibilité aux attaquants d'intercepter les messages transitant dans le réseau. Nous citons ci-dessous quelques attaques les plus connus :

1.9.3.1.a Attaque Sybil

Dans cette attaque, le nœud présente des identités multiples aux autres nœuds du réseau, créant ainsi des inconsistances dans les tables de routage des nœuds voisins. Ce qui permet de créer plusieurs routes passant par le nœud malicieux, qui ne sont en réalité qu'un seul chemin[24].

1.9.3.1.b Brouillage (jamming)

Le jamming est une attaque très connue qui s'en prend à la communication sans fil. En effet, vu la sensibilité du média sans fil au bruit, un nœud peut provoquer un déni de service en émettant des signaux à une certaine fréquence pour interférer avec les fréquences radio employées par les nœuds du réseau[25].

1.9.3.1.c Attaque du trou de ver (Wormhole attack)

Le terme Wormhole fait référence aux trous de ver en astronomie, qui sont des raccourcis entre deux points éloignés dans l'espace. Le principe est que l'attaquant utilise un chemin hors du réseau(ou un chemin virtuel par l'emploi de tunneling) pour faire passer les messages. Cette attaque requière plusieurs attaquants (au moins deux).

Chacun des deux attaquants va se placer non loin d'un des deux nœuds entre lesquels ils souhaitent intercepter le trafic. Ces deux attaquants disposent en plus de leur accès au réseau AD HOC, d'un lien direct physique (liaison radio) ou logique (un tunnel entre les deux). Ils vont employer cette liaison directe pour s'assurer d'être choisis comme route (le chemin le plus court), Ce chemin étant le plus court il sera emprunté par les messages entre les deux terminaux et donc intercepter par les attaquants[26].

1.9.3.1.d Attaque du trou noir (Blackhole)

L'attaque proposée par Ruiz, Friginal, David de-Andrés et Pedro Gil dans [27] est structurée en deux étapes successives (voir figure 1.8) :

1. Le nœud malveillant (M) se connecte à un réseau AD HOC vulnérable à l'attaque blackhole (figure 1.8(b)). Au départ, le nœud A(source) diffuse un message de demande de route (ROUTE_REQUEST)à tous les nœuds du réseau pour trouver un chemin vers le nœud destinataire D. Pour que M réalise l'attaque du trou noir il propose un lien de routage entre les dispositifs ciblés (A et D), puis il émet des messages compatibles(ROUTE_REPLY) avec le protocole de routage pour pousser à la fois A et D à choisir un tel lien pour leurs communications.
2. M effectue l'attaque (Figure 1.8(c)) en absorbant (ne retransmet pas) les paquets. Cet abandon de retransmission de paquets peut être sélectif (il ne touche qu'un type particulier de paquets) ou non (touche tous les types de paquets). De plus, M peut effectuer un autre type d'attaques en changeant simplement la façon dont il manipule les paquets interceptés (voir figure 1.8(c)). Par exemple, il peut créer de nouveaux paquets ou de modifier, de retarder ou de réorganiser ceux interceptés. Une fois que l'attaque a été injectée, son impact dans les communications et les applications fonctionnant à l'intérieur d'un réseau doit être évaluée. Cet impact peut, par exemple, conduire une application particulière à l'échec, dégrader les communications réseau, isoler les nœuds ou créer des boucles de routage.

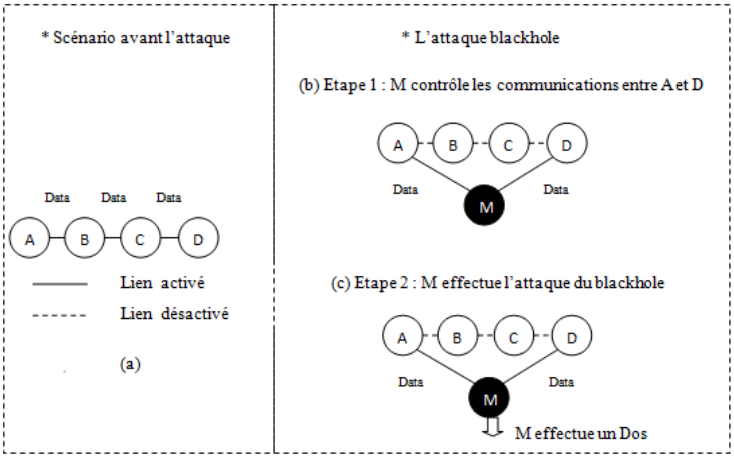

FIGURE 1.8: Attaque du blackhole

1.10 Conclusion

Dans ce premier chapitre, nous avons présenté un bref aperçu sur les notions de base de la sécurité informatique, les menaces et les attaques existantes, l'objectif de mettre en œuvre une politique de sécurité, ainsi l'architecture de la sécurité informatique et quelques techniques de défense, comme nous avons montré la sécurité dans les réseaux AD HOC. Cette étude nous a permis d'acquérir un ensemble de connaissances concernant les différentes méthodes utilisées pour parer aux menaces qui pèsent sur un réseau informatique, mais malheureusement aucune de ces solutions ne peut être à cent pour cent efficace comme solution générale pour les problèmes de sécurité. Car les réseaux informatiques souffrent toujours de nouvelles attaques et intrusions, les recherches dans ce domaine sont multipliées. Aujourd'hui les auteurs basent leurs recherches sur l'aspect mathématique et le résultat était de définir des méthodes formelles et qui sont applicables.

Chapitre 2

Etat de L'art sur l'Algèbre de Processus

2.1 Introduction

Aujourd'hui, la tâche de la détection des problèmes dans les systèmes informatiques est devenue de plus en plus coûteuse et délicate à cause de la complexité augmentée de ces derniers, en effet l'utilisation des méthodes formelles est incontournable pour la spécification et l'analyse de tels systèmes dans le but d'augmenter leur fiabilité. Pour cela, plusieurs langages formels ont été développés dans le but de décrire mathématiquement et sans ambiguïté des systèmes informatiques.

L'objectif de ce chapitre est l'étude d'une algèbre de processus pour la spécification des systèmes informatiques. L'intérêt des sujets traités est de faire avancer les résultats utiles et d'échapper à leurs limites.

2.2 Calcul et algèbre de processus

Ces vingt dernières années, une dizaines d'algèbres de processus ont été proposées, la majorité d'entre eux considèrent un système complexe et cela en fonction des actions élémentaires qu'il peut exécuter.

Une algèbre est définie par un ensemble d'éléments de base ainsi qu'un ensemble d'opérateurs permettant de construire des composantes complexes à partir des éléments basics. Parmi les algèbres de processus les plus connues dans le domaine de la spécification et la vérification des systèmes concurrents et les réseaux informatiques, nous trouvons : le CCS[28] et le π-Calcul[29] développées par Milner, le CSP[30] proposée par Hoare, l'ACP[31] introduite par Brookes et Roscoe et le calcul ambient[32] développée par Cardelli et Gordon.

2.2.1 CCS (Calculus of Communicating Systems)

Le calcul des systèmes communicants (CCS) développé par Milner[28] au cours des années 70, est un langage mathématique qui permet de décrire la concurrence des systèmes, pour déterminer avec précision les comportements qui seront vus ou expérimentés par un observateur.

2.2.1.1 Syntaxe du CCS

L'ensemble des éléments de base de cette algèbre contient des actions qui expriment l'envoi sur des canaux, notées généralement par a, b, c, etc., des co-actions qui expriment la réception sur des canaux, notées par $\bar{a}, \bar{b}, \bar{c}$, etc. Les actions a et $\bar{a}$ sont complémentaires. En plus, nous trouvons une action spéciale notée τ qui est utilisée souvent pour exprimer la synchronisation. Quant aux opérateurs de cette algèbre, ils sont les suivants : l'opérateur de préfixage noté par ".", l'opérateur du choix noté par "+", l'opérateur de parallélisme noté par "|", l'opérateur de renommage noté par "[]" et l'opérateur de restriction noté par "\"[31].

Nous notons par $\sum$ l'ensemble des actions et $\overline{\sum}$ l'ensemble des co-actions, on obtient alors :Act=$\sum \cup \overline{\sum} \cup \tau$. Soit a une action dans Act, la syntaxe du CCS peut être décrite par la BNF (Backus-Naur Form) donnée sur le tableau 2.1.

P : : =	0	Processus inactif
	|	a.P	Préfixage
	|	X=P	Définition
	|	P1+P2	Choix
	|	P1 | P2	Parallélisme
	|	P[f]	Renommage
	|	P \ L	Restriction

TABLE 2.1: Syntaxe du CCS

2.2.1.2 Sémantique opérationnelle du CCS

D'une manière informelle, la sémantique d'un processus du CCS peut être décrite comme suit :

- **Processus inactif** : c'est le processus le plus élémentaire, il correspond au processus qui n'exécute aucune action.
- **Prefixage a.P** : c'est le processus qui exécute l'action « a » et il se comporte par la suite comme P.
- **Définition X = P** : permet de définir des processus qui ont un comportement infini. Par exemple, X = a.X est le processus qui fait tout le temps a.
- **Choix P1 + P2** : c'est un processus qui peut se comporter soit comme P1 soit comme P2. Ce choix peut être déterministe ou indéterministe.
- **Parallélisme P1 | P2** : P1 et P2 peuvent évoluer séparément ou de manière synchronisée pour former une action silencieuse. La seule possibilité permettant à deux composantes d'un processus d'évoluer en même temps est via une communication.
- **Renommage P[f]** : via une fonction de renommage f, les actions d'un processus peuvent être renommées pour en définir un nouveau processus. Par ailleurs, une fonction de renommage f doit respecter les conditions suivantes
  $f(\bar{}a) = \overline{f(a)}$ et $f(\tau) = \tau$.
- **Restriction P\L** : un processus se comporte comme P à l'exception qu'il ne peut pas exécuter des actions qui sont dans l'ensemble L. Cet opérateur spécifie les actions internes à un processus.

2.2.2 CSP (Communicating Sequential Processes)

En programmation concurrente, Communicating Sequential Processes (CSP) est une algèbre de processus permettant de modéliser l'interaction des systèmes.

CSP intègre un mécanisme de synchronisation basé sur le principe du rendez-vous ; combinant ce mécanisme à une syntaxe simple et concise, CSP permet alors l'implémentation rapide des paradigmes classiques de la concurrence, tels que producteurs/consommateurs ou lecteurs/écrivains. Cependant CSP n'est pas un langage de programmation complet.

CSP fut décrit en premier par Hoare [34] en 1978, mais à depuis évolué de façon substantielle. CSP a été mis en pratique industriellement comme un outil de spécification formelle de l'exécution concurrente des systèmes variés. Les éléments de la syntaxe du

CSP sont les suivants :

- **Préfixe** : pour spécifier les événements qui précèdent un certain processus.
- **Récurrence** : pour représenter les tâches répétitives.
- **Choix** : pour exprimer un comportement alternatif.
- **Récursion mutuelle** : pour accueillir de multiples solutions.

Soient s et t des traces et soit A un ensemble d'événements. Les opérations suivantes sont définies sur les traces :

- La concaténation $s \frown t$: pour la construction de trace d'événements individuels.
- La restriction $t \upharpoonright$ A : pour extraire le sous-ensemble d'événements en A à partir d'une trace t.
- La tête s_0 et la queue s' : pour identifier les premiers et derniers éléments d'une trace.
- L'étoile A* : pour désigner toutes les traces finies avec les symboles de A.
- La commande $s \leq t$: pour définir une relation d'ordre entre les traces s et t.
- La longueur $\sharp t$: pour compter le nombre d'événements de la trace t.

Parmi les autres opérations des traces le symbole de changement, l'entrelacement, la sélection, l'inversion et la composition. CSP a consacré les opérateurs pour exprimer à la fois le choix déterministe et non déterministe, et c'est la seule algèbre de processus bien établie que les autres.

Le but du CSP tel que déclaré par son auteur est d'être la plus simple théorie mathématique qui fournit une assistance claire pour le programmeur dans ses tâches de spécification, de conception, de mise en œuvre, de vérification et de validation des systèmes informatiques complexes[34].

2.2.3 π-Calcul

Le π-calcul a été introduit par Milner [35, 36, 37] , Parrow [38] et Walker[39]. Ce calcul se base principalement sur le CCS, sans toute fois en être une extension. Il a pour vocation première de modéliser des échanges de messages entre des programmes s'exécutant en parallèle. Les messages s'échangent sur des canaux, qui peuvent être publiques ou privés. Le π-calcul améliore considérablement l'aspect communication,

déjà présent dans le CCS, en permettant le passage non seulement de valeur, mais également de canal de communication, voire même de processus entier. Cette nouveauté amène une notion de mobilité qui n'était pas présente dans le CCS.

2.2.3.1 Syntaxe du π-calcul

La syntaxe du π-calcul est similaire à celle du CCS. Elle ne diffère que par ses actions qui donnent la possibilité d'envoyer et de recevoir des noms de canaux. La syntaxe du π-calcul est donnée dans le tableau 2.2 :

Préfixes	α :: =	$\bar{a}x$	Emettre x sur un canal α
	|	a(x)	Recevoir x sur un canal α
	|	τ	Action silencieuse
Agents	P,Q:: =	0	Processus nul
	|	α.p	Préfixe
	|	P+Q	Choix
	|	P | Q	Parallélisme
	|	if x=y then P else Q	Tests d'égalité
	|	(vx)P	Restriction

TABLE 2.2: Syntaxe du π-calcul

- L'égalité if x = y then P else Q : un processus peut tester l'égalité entre deux noms. Le processus se comporte comme P dans le cas où x et y sont égaux sinon, il se comporte comme Q.
- La restriction (vx)P : elle joue un rôle analogue à l'opérateur de CCS. Ainsi, le processus (vx)P se comporte comme P, mais le nom x est local au P, c'est à dire que P n'utilise pas son nom x pour communiquer avec l'environnement externe.

2.2.3.2 Sémantique opérationnelle du π-calcul

La majorité des règles de la sémantique opérationnelle du π-calcul ressemblent à celles du CCS sauf, la règle d'ouverture et les fonctions *bn* et *fn* qui apparaissent dans la règle de parallélisme et de restriction. Les appellations *bn* et *fn* proviennent de l'anglais bound names et free names.

Les fonctions bn(P) et fn(P) représentent respectivement les ensembles des noms liés et libres dans le processus P. Un préfixe a un sujet a et un objet x, l'objet est dit libre dans le préfixe d'émission et lié dans le préfixe de réception $a(x)$.P. L'opérateur de restriction (vx)P lie également x à P. Les formules suivantes définissent ces deux fonctions :

$$\begin{array}{ll} fn(ax.P) = a, x \cup fn(P) & bn(ax.P) = bn(P) \\ fn(a(x).P) = a \cup fn(P) & bn(a(x).P) = x \cup bn(P) \\ fn((vx)P) = fn(P) & fn((vx)P) = x \cup bn(P) \end{array}$$

La règle d'ouverture permet d'éliminer la restriction d'un canal à un processus, et transforme du coup une action d'envoi libre $\xrightarrow{\bar{a}x}$ en une action d'envoi liée $\xrightarrow{\bar{a}vx}$. La sémantique opérationnelle du π-calcul est représentée dans le tableau 2.3 :

Équivalence	$\frac{P' \equiv P, P \xrightarrow{\alpha} Q, Q \equiv Q'}{P' \xrightarrow{\alpha} Q'}$
Préfixe	$\frac{}{\alpha.P \xrightarrow{\alpha} P}$
Choix	$\frac{P \xrightarrow{\alpha} P'}{P+Q \xrightarrow{\alpha} P'}$
Concordance	$\frac{P \xrightarrow{\alpha} P'}{if x=y then P \xrightarrow{\alpha} P'}$
Non concordance	$\frac{P \xrightarrow{\alpha} P'}{if x \neq y then P \xrightarrow{\alpha} P'}\ x \neq y$
Entrelacement	$\frac{P \xrightarrow{\alpha} P'}{P \mid Q \xrightarrow{\alpha} P' \mid Q}\ bn(\alpha) \cap fn(Q) = \emptyset$
Communication	$\frac{P \xrightarrow{\alpha(x)} P', Q \xrightarrow{\bar{a}u} Q'}{P \mid Q \xrightarrow{\alpha} P\{u/x\} \mid Q'}$
Restriction	$\frac{P \xrightarrow{\alpha} P'}{(vx) \xrightarrow{\alpha} (vx)P'} x \notin bn(a) \cup fn(a)$
Ouverture	$\frac{P \xrightarrow{\bar{a}x} P'}{(vx)P \xrightarrow{\bar{a}vx} P'}\ a \neq x$

TABLE 2.3: Sémantique opérationnelle du π-calcul

2.2.4 Le Calcul Ambient

Le calcul ambient a été développé par Cardelli et Gordon [40] au cours des années 90. L'objectif de leurs travaux était d'offrir un calcul qui permettrait d'exprimer tous les aspects de la mobilité, en visant principalement les agents mobiles et le code mobile. Un ambient est un espace (un réseau, une machine, une page web, etc.) borné ayant un intérieur et un extérieur, dans lequel peuvent se dérouler des calculs, il possède un nom unique non modifiable. Ce nom est utilisé pour contrôler les accès à l'ambient, à l'intérieur d'un ambient, on peut trouver des processus concurrents, ou bien une collection d'ambients. Ainsi chaque ambient possède une collection d'agents locaux, ces derniers, plus simple que les ambients, en définissent le comportement, ils peuvent par exemple permettre à l'ambient de pénétrer à l'intérieur d'un autre ambient. Nous décrivons maintenant la syntaxe du calcul ambient et sa sémantique opérationnelle.

2.2.4.1 Syntaxe du calcul ambient

La syntaxe du calcul ambient est comme indiquée dans le tableau 2.4.

- Le processus "0" représente le processus inactif comme dans le CCS et le π-calcul.

	(a) Processus		
P	:: =	0	Inactivité
	|	(vn)P	Restriction
	|	P|P'	Composition parallèle
	|	!P	Réplication
	|	n[p]	Ambient
	|	M.P	Action
	|	⟨M⟩	Sortie d'une capacité
	|	(x).P	Entrée d'une capacité
	(b) Capacités		
	|	in N	Entrer dans N
	|	out N	Sortir de N
	|	open N	Ouvrir N
	|	ε	Capacité nulle
	|	M.M'	Chemin d'accès
	(c) Noms		
	|	n	Nom
	|	≪n≫	Sortie d'un nom
	|	((n)).P	Entrée d'un nom

Table 2.4: Syntaxe du calcul ambient

- La restriction de n à P, notée(vn)P, et la composition parallèle notée par le symbole "|" ont les mêmes significations que celle du π-calcul.
- La notation n [P] dénote un processus P exécuté dans un ambient nommé n.
- L'opérateur de réplication " ! " permet d'obtenir autant de duplications parallèles souhaitées d'un processus donné. Le processus !P est donc équivalent à P| !P.
- Le symbole M.P représente un processus qui commence par exécuter la capacité M puis il continue comme P.
- Les symboles ⟨M⟩ et (x)P expriment une communication qui permet de passer d'une capacité M d'un processus émetteur ⟨M⟩ à un processus récepteur (x)P parallèle et voisin (localisé dans le même ambient).
- La capacité ε est la capacité vide.
- La capacité M.M' signifie la concaténation de deux capacités M et M'.

Les capacités *in* N, *out* N et *open* N : *in* N fait entrer l'ambient conteneur dans un ambient parallèle N, *out* N fait sortir l'ambient conteneur de son ambient parent N, tandis que *open* N élimine les frontières de l'ambient fils N.

2.2.4.2 Sémantique opérationnelle du calcul ambient

La sémantique opérationnelle du calcul ambient est définie via deux relations ; la congruence entre processus $\equiv$ et des règles de réduction $\rightarrow$ comme le montre le tableau 2.5 et le tableau 2.6 respectivement.

Dans le calcul ambient, l'opérateur $\equiv$ représente un processus ayant la même structure interne, les règles régissant ces évolutions sont appelées règles de congruence structurelle. Quant à l'opérateur $\rightarrow$ il est utilisé pour marquer l'évolution d'un processus en un autre processus, les règles régissant ces évolutions sont appelées règles de réduction. Ainsi, lorsque l'on note P $\rightarrow$Q, cela indique que le processus P évolue pour en devenir le processus Q. La relation de congruence est définie par les axiomes et les règles du tableau 2.5. Ils assurent que la relation de congruence est une relation d'équivalence, que la composition parallèle est commutative et associative avec le processus inactif, ils décrivent aussi le comportement de la réplication.

$P \equiv P$	$P \mid Q \equiv Q \mid P$
$P \equiv Q \Rightarrow Q \equiv P$	$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$
$P \equiv Q, Q \equiv R \Rightarrow P \equiv R$	$!\, P \equiv P \mid !\, P$
$P \equiv Q \Rightarrow (vn)Q \equiv (vn)P$	$P \mid 0 \equiv P$
$P \equiv Q \Rightarrow P \mid R \equiv Q \mid R$	$(vn).0 \equiv 0$
$P \equiv Q \Rightarrow !\, P \equiv !\, Q$	$!\, 0 \equiv 0$
$P \equiv Q \Rightarrow n[Q] \equiv n[P]$	$\varepsilon.P \equiv P$
$P \equiv Q \Rightarrow (x).Q \equiv (x).P$	$(M.M').P \equiv M.M'.P$
$P \equiv Q \Rightarrow M[Q] \equiv M[P]$	$P \equiv Q \Rightarrow M.P \equiv M.Q$

TABLE 2.5: Règles de congruence structurelle

La relation de réduction est définie par les axiomes et les règles du tableau 2.6, ces règles illustrent d'une part l'emploi des capacités, et spécifient d'autre part que la congruence structurelle peut être utilisée pour réarranger les expressions des processus

ambients, les restrictions, les processus ambients ainsi que les compositions parallèles.

n[in m.P | Q] | m[R] → m[n[P | Q] | R]	P ≡ P',P' → Q',Q' ≡ Q ⇒ P ≡ Q
m[n[out m.P | Q] | R] → n[P | Q] | m[R]	P → Q ⇒ (vn)(P) → (vn)(Q)
open n.P | n[Q] → P | Q	P → Q ⇒ n[P] → n[Q]
⟨M⟩ | (x).P → P{M →x}	P → Q ⇒ P | R → Q | R

TABLE 2.6: Règles de réduction

2.3 Contrôle d'accès avec les ambients mobiles

Ces dernières années, plusieurs travaux de recherche ont utilisé le calcul ambient pour répondre à un certain nombre de questions liées au contrôle d'accès. Les solutions sont proposées sous la forme : d'informations et d'analyse des flux de contrôle, les ambients en boite surveillés, les safe ambients et les calculs de spécification. Nous présentons dans cette section les travaux faits dans ces domaines.

2.3.1 Analyse de flux de contrôle

Le calcul ambient est utilisé dans cette approche comme base pour une analyse de flux d'informations. Le facteur déterminant de cette méthode est donné par l'aspect de mobilité des ambients. Des politiques sont employées pour déterminer quels sont les flux autorisés, par conséquent établir de façon formelle si une certaine propriété de flux est maintenue. Deux analyses sont présentées : l'analyse de flux d'informations de Cortesi et Focardi[41] et la validation de pare-feu de Nielson et al[42].

2.3.1.1 Analyse de flux d'informations

Cortesi et Focardi[41] se sont concentré sur la sécurité à plusieurs niveaux, particulièrement sur une politique de sécurité de contrôle d'accès obligatoire, cette dernière exige que chaque entité est liée à un niveau de sécurité (seulement deux niveaux sont considérés : haut et bas), et l'information ne peut circuler que du niveau bas au niveau haut. Deux règles d'accès sont imposées : No Read Up (une entité de bas niveau ne

peut pas lire une information d'une entité de haut niveau), No Write Down (une entité de haut niveau ne peut pas écrire des informations sur une entité de bas niveau).

Les auteurs considèrent la sécurité de flux d'information dans le cas des ambients mobiles, le scénario envisagé est que le code peut migrer d'un niveau de sécurité à un autre et cela complique la capture de fuites d'informations dans le système. La propriété de flux d'information est définie en fonction de la possibilité de déplacer un ambient confidentiel en dehors d'une frontière de sécurité en utilisant des canaux cachés (chemin virtuel entre l'entité de bas niveau vers l'entité de haut niveau).

Des étiquettes spéciales sont attribuées aux capacités internes et externes des ambients contenant des données sensibles (ambients de frontières). Le processus de vérification consiste à prouver qu'une capacité étiquetée est suffisante pour garantir l'absence des flux d'informations non désirés. L'analyse deviendrait très complexe et encombrante dans le cas d'une sécurité multi-niveaux ou d'une combinaison de propriétés de sécurité.

2.3.1.2 Validation de pare-feu

Dans [42], Nilson et al ont utilisé le calcul ambient pour spécifier un pare-feu et les agents qui tentent de le franchir, le modèle de pare-feu est tiré de l'article original sur les ambients mobiles de Cardelli et Gordon. Le principe est qu'un pare-feu correctement spécifié ne doit pas permettre l'entrée aux attaquants qui n'ont pas les mots de passe requis. Pour cela les auteurs ont développé une analyse de flux qui peut être utilisée pour concevoir un test permettant de déterminer si un pare-feu donnée est fiable.

Le modèle de pare-feu utilisé par Nielson et ses collaborateurs, ne représente pas exactement la façon dont les pare-feu agissent. La complexité de l'analyse est loin d'être idéale pour la mise en œuvre dans les machines d'exploitation en temps réel telles que les pare-feu.

2.3.2 Les safe ambients et approches dérivées

Le calcul des ambients mobiles a inspiré un certain nombre d'autres chercheurs de le développer. Une approche différente appelée safe ambients est adressée par Degano, Lévi, Bugliesi, Sangiorgi et al[44, 45, 46, 43].

2.3.2.1 Safe ambients

Un nouveau calcul appelé safe ambients (ambients de sécurité) est mis en place par Levi et Sangiorgi[44, 45], ils examinent l'interférence des processus qui est l'un des problèmes les plus complexes liés à la concurrence. L'idée est de modifier les actions de mouvement de Cardelli et Gordon [40] de façon à mieux contrôler les interférences de processus ; en effet les auteurs ont ajouté trois nouvelles primitives appelées co-capacités $\overline{in}$ n, $\overline{out}$ n et $\overline{open}$ n, celles-ci résume que : un mouvement de processus n'aura lieu que si les deux parties participantes à l'interaction se sont mises d'accord. Ce qui permet d'écrire des programmes plus robustes et facilite de prouver les propriétés comportementales des processus. Cette approche peut être améliorée avec un système de type qui assure le contrôle de la mobilité et supprime toutes les interférences dangereuses.

2.3.2.2 Sécurité des safe ambients

La sécurité des safe ambients est définie par Bugliesi et Castagna dans [46], ils ont proposé un système de type qui permet l'expression et la vérification des comportements invariants des ambients. Ce système est capable de capturer à la fois tous les processus (explicite et implicite) qui s'exécutent et les comportements des ambients.

Les auteurs ont défini dans le système de type du SSA(Secure Safe Ambient) des algorithmes de vérification et de reconstruction de type et un langage pour exprimer des propriétés de sécurité.

Par conséquent, les types s'appliquent à la fois au comportement immédiat des processus et au comportement des résidus du traitement, couvrant toute évolution possible des processus dans un contexte donné. Cette caractéristique particulière permet au système de type du SSA de détecter des attaques de sécurité potentielles comme les chevaux de Troie et d'autres combinaisons d'agents malveillants.

2.3.3 Les ambients en boîte surveillés

Ferrari et al [47], proposent une extension du calcul ambient, en prenant en compte le concept de sécurité et de la politique de coordination, en attachant un gardien pour

chaque ambient. Leur modèle formel, appelé les ambients en boite surveillés, prend en charge la spécification de plusieurs politiques de sécurité dynamiques en séparant les mécanismes informatiques (primitives de communication et de la mobilité) des autorités de politique. Les gardiens définissent un contexte de sécurité locale et contrôlent l'activité des processus et des sous-ambients ainsi que l'interaction avec l'environnement externe. Ils mettent en œuvre l'ensemble d'autorisations qui peuvent être accordées aux agents entrants, sortants, et de communiquer dans un environnement surveillé. Les gardiens ont aussi des capacités de coordination et ils peuvent collaborer avec succès pour une propagation efficace d'un changement ou d'une modification de politique dans un environnement. Nous trouvons cette approche très complexe et difficile de l'implémentée dans le monde réel, lorsque par exemple le nombre des sous-ambients devient important.

2.3.4 Les ambients contrôlés

Teller et al [48] se sont intéressé à l'analyse du contrôle de ressources, où ils considèrent une ressource comme étant une entité pouvant être acquise, utilisée, puis libérée. Les auteurs ont développé une extension du calcul ambient nommée les ambients contrôlés ; qui est adaptée à modéliser certaines attaques comme les dénis de service. Dans un ambient contrôlé, chaque déplacement est sujet à un contrôle effectué sur trois parties différentes : l'ambient qui se déplace, l'ambient qui reçoit un nouveau sous-ambient et l'ambient qui laisse partir un de ses sous-ambients. Ces contrôles se font grâce aux cinq nouvelles capacités ajoutées à l'algèbre :

- $\overline{in} \uparrow$ m : permet à un ambient « m » provenant d'un sous-ambient d'entrer.
- $\overline{in} \downarrow$ m : permet à un ambient « m » provenant d'un ambient parent d'entrer.
- $\overline{out} \uparrow$ m : permet à « m » de quitter l'ambient où il se trouve en sortant de celui-ci.
- $\overline{out} \downarrow$ m : permet à « m » de quitter l'ambient où il se trouve en entrant dans un de ses sous-ambients.
- $\overline{open}$ {m,h} : permet à l'ambient parent « h » d'ouvrir l'ambient courant « m ».

Les caractéristiques du contrôle de ressources requises pour modéliser les attaques de déni de service sont intégrés dans les ambients à l'aide de deux nouveaux paramètres

appelés capacité et poids. Ces paramètres déterminent respectivement : combien de ressources un ambient offre à ses sous-ambients et combien de ressources un ambient requiert de ses ambients parents. Les auteurs ont proposé un système de type qui permet de garantir statiquement le respect d'une politique de contrôle de ressources, en introduisant des jugements pour les différents types (les ambients, les processus et les messages) et des règles de communication et d'allocation de ressources.

Les auteurs affirment que leur model est «plus raisonnable» et «plus réaliste» que d'autres approches, comme les safe ambients, mais la construction de ce modèle est assez complexe et il est seulement utile dans le cas des attaques de déni de service. Les politiques de ressources se consistent uniquement sur la déclaration de disponibilité, plutôt que sur les exigences du contrôle d'accès (de sécurité).

2.4 Conclusion

Dans ce chapitre, nous avons présenté des exemples de l'algèbre de processus comme le CCS, CSP, π-calcul et le calcul ambient, permettant de spécifier des systèmes informatiques, où nous nous sommes approfondis sur le calcul ambient, ces algèbres permettent de modéliser le fonctionnement des processus en termes des actions qu'ils peuvent exécuter.

Ainsi, nous avons présenté les approches de contrôle d'accès qui utilisent une variante de calcul ambient(analyse de flux de contrôle, les safe ambients, les ambients contrôlés et les ambients surveillés..etc.), qui impliquent des constructions élaborés pour vérifier des propriétés relativement simple, ce qui rend l'évolutivité très limité. Dans le chapitre suivant, nous étudions une approche qui permet de spécifier la sécurité des systèmes et réseaux informatiques, afin de l'appliquer aux réseaux AD HOC.

Chapitre 3

Application de l'Approche RSC aux Réseaux AD HOC

3.1 Introduction

Les réseaux sans fil sont par nature sensibles aux problèmes de sécurité. L'intrusion sur le support de transmission est plus facile que pour les réseaux filaires et il est possible de mener facilement des attaques de déni de service (DoS).

L'objectif de ce chapitre est de sécurisé un réseau AD HOC contre l'attaque de déni de service causé par un trou noir, pour cela nous appliquons l'approche RSC afin d'obtenir un système plus sécurisé.

Le chapitre est organisé comme suit : la section 3.2, présente le calcul du renforcement de la politique de sécurité (RSC). La section 3.3 montre la discussion sur l'approche RSC et en fin la section 3.4 applique l'approche RSC sur un réseau AD HOC en spécifiant un système vulnérable à l'attaque blackhole.

3.2 Calcul du renforcement de la politique de sécurité (approche RSC "Renforcement Security Calculus")

Dans cette section nous présentons la syntaxe et la sémantique du calcul de [49, 50], ce dernier est adapté pour décrire l'interaction avec un intrus potentiel ; les auteurs modélisent les composants du système en termes de processus et interactions de processus.

3.2.1 Syntaxe

Elle définit les modules requis pour un renforcement de sécurité dans un système informatique via des clefs de sécurité et elle est présentée dans le tableau 3.1. Soit N l'ensemble des noms, K l'ensemble des clefs, et V l'ensemble des variables. Dans cette approche chaque processus est emboité à l'intérieur d'un ambient qui est protégé via deux clefs de sécurités k et k' ce qui dénote l'accès contrôlé aux ressources. Les clefs sont saisies comme clefs d'entrée et de sortie (e et s, respectivement). Par conséquent, des processus doivent connaitre la clef appropriée afin d'entrer dans un ambient.

Les ambients nouvellement crées contiennent par défaut deux clefs de même valeur δ connue par tous les autres processus ; le symbole δ définit une clef publique.

Dans la suite nous dénotons $\mathcal{P}$ l'ensemble de tous les processus qui peuvent être exprimés par le calcul de [49, 50]. L'opérateur de concaténation "." décrit le comportement séquentiel d'un processus. Le terme ACTION est employé pour décrire l'exécution des capacités de processus, et ci-dessous nous présentons l'ensemble $\mathcal{P}$:

- **Inactivité** : 0 est un processus qui n'exécute rien.
- **Composition parallèle** :P | Q se rapporte à l'exécution parallèle des processus P et Q.
- **Réplication** : !P est un processus qui dénote la réplication illimitée du processus P. c'est un opérateur commutatif et associatif.
- **Ambient protégé** : ${}^{k,k'}_{n}$[P] dénote un ambient n qui contient une ressource P protégée par deux clefs k et k'.
- **Action** : a.P présente le comportement séquentiel d'un processus comme séquence : il emploie d'abord la capacité « a » puis se comporte comme P.

Les auteurs ont combiné les notions d'ambient protégé et d'action dans la séquence d'ambient, ce qui dénote le fait qu'une action peut se produire à l'intérieur d'un ambient protégé.

Les sous-catégories suivantes se rapportent à des capacités de processus :

- **Demande de clef** : $req^{x}_{n,t}$ permet à un processus de faire une demande de clef d'accès k ou de sortie k', selon la valeur du paramètre t.
- **Publication de clef** : $pub^{x}_{n,t}$ se rapporte à la publication de clefs d'accès ou de

sortie selon la valeur du paramètre t.

- **Mouvement** : mov_n^k se rapporte aux capacités d'un processus pour se déplacer en utilisant la clef d'ambient appropriée.
- **Exploration** : exp permet la présentation des processus dynamiques qui peuvent choisir d'une manière non-déterministe ce qu'est leur prochaine action.
- **Protection** : $prot_n^{k,k'}$ permet de renforcer la protection des ressources à l'intérieur d'un ambient n avec la paire ordonnée k et k' se composant d'une clef d'accès et d'une clef de sortie.
- **Unprotection** : $unprot_n^{k,k'}$ permet d'enlever la protection de l'ambient n et de la rendre par défaut δ, δ.
- **Choix compositionnel** : a $\oplus$ b permet à l'évolution d'un processus d'être définie comme choix entre toute combinaison possible des actions a et b
- **Choix non déterministe** : a $\sqcap$ b permet à l'évolution d'un processus d'être définie comme choix entre deux processus composants, mais ne permet pas à l'environnement n'importe qu'elle contrôle sur des processus composants.

Il existe deux types de mouvement selon le paramètre k : mouvement d'accès et mouvement de sortie . Le mouvement d'accès permet à un processus d'entrer dans un ambient n protégé par la clef d'accès k. Le mouvement de sortie permet à un processus de sortir d'un ambient n protégé par la clef de sortie k'. La représentation séquentielle des processus (actions suivies d'autres actions) exprime le fait que la toute première action doit se produire avant que la prochaine soit exécutée. Un ordre prédéfini des actions prêtera au processus un comportement prévisible. Ces processus sont considérés comme des processus réguliers. Cependant, il n y'a pas des processus qui n'ont pas un comportement prévisible. L'opérateur exp est utilisé pour imiter le comportement dynamique d'un intrus. Les divers aspects du comportement de processus sont présentés dans la section 3.2.2.

n	$\in$	N	Nom
x	$\in$	V	Variable
k, k'	$\in$	K$\cup\{\delta\}$	Clefs de sécurité
t	$\in$	$\{e, s\}$	Type de clef
P,Q	::=		Processus
	|	0	Inactivité
	|	P | Q	Composition parallèle
	|	!P	Réplication
	|	$^{k,k'}_{n}$[a.P]	Séquence d'ambient
a,b	::=		Capacités de processus
	|	$req^{x}_{n,t}$	Demande de clefs
	|	$pub^{x}_{n,t}$	Publication de clefs
	|	mov^{k}_{n}	Mouvement
	|	exp	Exploration
	|	$prot^{k,k'}_{n}$	Protection
	|	$unprot^{k,k'}_{n}$	Unprotection
	|	a $\oplus$ b	Choix compositionnel
	|	a $\sqcap$ b	Choix non-déterministe

TABLE 3.1: Syntaxe de l'approche RSC

3.2.2 Sémantique

Les deux composants de la sémantique opérationnelle du calcul sont : la congruence structurelle notée par $\equiv$, et la relation de réduction notée par $\rightarrow$. Le tableau 3.2 détaille la congruence structurelle en tant qu'équivalence de processus dans le contexte du calcul. La congruence séquence d'ambient est une extension de la séquence d'équivalence de processus aux processus protégés. Le zéro parallélisme énonce l'élément neutre du calcul. Les itérations sont dénotées par la réplication, qui peut être employée pour modéliser un service (tel que la publication de clefs) ou une tentative répétée d'accéder à un domaine protégé. Le choix compositionnel décrit le comportement potentiel de l'intrus : soit a et b deux capacités, a$\oplus$b signifie que le processus peut continuer soit comme une action a ou comme b, ou en tant que séquence de deux capacités a.b ou

b.a. Le choix d'exécution illustre comment le choix non-déterministe des actions est effectué pour l'exécution de processus.

La relation de réduction définie dans le tableau 3.3 capture le raccordement entre le comportement des processus, les capacités des processus et la protection d'ambient. Les règles (1) et (2) sont triviales. La règle (3) décrit le fait qu'un processus peut entrer dans un ambient protégé. La capacité de sortir d'un ambient protégé est présentée par la règle (4). Les règles (5) et (6) dénotent le niveau de la protection offert par la clef publique δ, les tentatives d'accéder ou de sortir d'un ambient protégé par δ comme clef d'accès ou de sortie, respectivement, sont réussies indépendamment de la clef utilisée.

$P \equiv P$	Réflexivité
$P \equiv Q \Rightarrow Q \equiv P$	Symétrie
$P \equiv Q \wedge Q \equiv R \Rightarrow Q \equiv R$	Transitivité
$P \equiv Q \Rightarrow Q \mid R \equiv P \mid R$	Parallélisme
$P \equiv Q \Rightarrow {}^{k,k'}_{n}[a.P] \equiv {}^{k,k'}_{n}[a.Q]$	Séquence d'ambient
$P \equiv 0 \equiv P$	Zéro parallélisme
$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$	Associativité
$P \mid Q \equiv Q \mid P$	Commutativité
$!P \equiv P \mid !P$	Réplication parallèle
$P \equiv Q \Rightarrow !P \equiv !Q$	Réplication
$!0 \equiv 0$	Zéro réplication
$a \oplus b \equiv a \sqcap b \sqcap a.b \sqcap b.a$	Choix compositionnel
$a \oplus b \equiv b \oplus a$	Commutativité choix compositionnel
$(a \oplus b) \oplus c \equiv a \oplus (b \oplus c)$	Associativité choix compositionnel
$a \sqcap b \equiv b \sqcap a$	Commutativité choix non déterministe
$(a \sqcap b) \sqcap c \equiv a \sqcap (b \sqcap c)$	Associativité choix non déterministe
$(a \sqcap b).P \equiv a.P \sqcap b.P$	Choix d'exécution

TABLE 3.2: Règles de congruence structurelle

Le symbole « _ » désigne n'importe qu'elle valeur de la clef à employer par les capacités *mov*. Une demande de la clef appropriée du service approprié de publication a comme conséquence une communication de la valeur de la clef (7). La règle (8) décrit le renforcement des clefs de sécurités. L'action $prot_n^{k,k'}$ peut seulement être exécutée sur des ambients protégés avec une paire de clefs publiques. Afin de changer les clefs,

l'action $unprot_n^{k,k'}$ est nécessaire, comme illustré par la règle (9). La paire de clefs indiquées par le $unprot_n^{k,k'}$ devrait assortir la paire réelle de protection de l'ambient. Le résultat d'un *unprot* réussi est un ambient protégé avec une paire de clefs publiques. La règle (10) décrit le comportement de l'intrus. Elle modélise la manière dont le processus d'intrus peut employer un mouvement d'accès ou de sortie s'il aboutie à n'importe quel privilège pour un certain scénario d'attaque. Cette relation s'ajoute à la connaissance de l'intrus et lui donne de nouvelles capacités de mouvement.

La compréhension de ce qui peuvent être prévus de l'intrus est cruciale pour assurer la protection proportionnée à un système. Le calcul de [49, 50] permet d'identifier des menaces potentielles en combinant le $\oplus$ de mov_n^k et de l'expression (exp).P de la règle (10) dans le tableau 3.3 avec le choix compositionnel et le choix d'exécution de la relation de congruence structurelle présentée dans le tableau 3.2. Nous obtenons :

$(mov_n^k \oplus exp).\mathrm{P} \equiv (mov_n^k.\mathrm{P}) \sqcap (exp.\mathrm{P}) \sqcap (mov_n^k.exp.\mathrm{P}) \sqcap (exp.mov_n^k.\mathrm{P})$

Par conséquent, l'intrus a un choix pour faire une des quatre options suivantes : utiliser la clef et cesser d'explorer, ignorer la clef et continuer l'exploration, utiliser la clef puis continuer d'explorer et, en conclusion, continuer d'explorer et utiliser la clef à un futur moment. Si l'intrus choisit, par exemple, la troisième option ($mov_n^k.exp.\mathrm{P}$), alors le résultat est un mouvement intérieur ou extérieur d'un ambient n, dépendant si t prend la valeur e ou s. C'est les scénarios illustrés dans a) et b) du tableau 3.4. Les scénarios c) et d) illustre le fait que si la clef publique est employée pour la protection d'ambient, le processus d'intrus peut entrer ou sortir comme il veut.

P' $\rightarrow$ Q' if P' $\equiv$ P,P $\rightarrow$ Q , Q $\equiv$ Q'	(1)
P $\mid$ R $\rightarrow$ Q $\mid$ R if P $\rightarrow$ Q	(2)
mov_n^k.P $\mid {}_n^{k,k'}$[Q] $\rightarrow {}_n^{k,k'}$[P $\mid$ Q]	(3)
${}_n^{k,k'}[mov_n^{k'}$.P $\mid$ Q] $\rightarrow$ P $\mid {}_n^{k,k'}$[Q]	(4)
mov_n^-.P $\mid {}_n^{\delta,k'}$[Q] $\rightarrow {}_n^{\delta,k'}$[P $\mid$ Q]	(5)
${}_n^{k,\delta}[mov_n^-$.P $\mid$ Q] $\rightarrow$ P $\mid {}_n^{k,\delta}$[Q]	(6)
$req_{n,t}^x$.P $\mid (pub_{n,t}^k$.Q) $\rightarrow$ P$[x \leftarrow k] \mid (pub_{n,t}^k$.Q)	(7)
$prot_n^{k,k'}$.P $\mid {}_n^{\delta,\delta}$[Q] $\rightarrow$ P $\mid {}_n^{k,k'}$[Q]	(8)
$unprot_n^{k,k'}$.P $\mid {}_n^{k,k'}$[Q] $\rightarrow$ P $\mid {}_n^{\delta,\delta}$[Q]	(9)
exp.P $\mid (pub_{n,t}^k$.Q) $\rightarrow (mov_n^k \oplus exp)$.P $\mid (pub_{n,t}^k$.Q)	(10)

TABLE 3.3: Règles de réduction

a) exp.P $\mid (pub_{n,t}^k$.Q) $\mid {}_n^{k,k'}$[R] $\rightarrow (pub_{n,e}^k$.Q) $\mid {}_n^{k,k'}$[R $\mid$ exp.P]
b) ${}_n^{k,k'}[exp$.P $\mid (pub_{n,s}^{k'}$.Q) $\mid$ R] $\rightarrow$ exp.P $\mid {}_n^{k,k'}[(pub_{n,s}^{k'}$.Q) $\mid$ R]
c) exp.P $\mid {}_n^{\delta,k'}$[R] $\rightarrow {}_n^{\delta,k'}$[R $\mid$ exp.P]
d) ${}_n^{k,\delta}[exp$.P $\mid$ R] $\rightarrow$ exp.P $\mid {}_n^{k,\delta}$[R]

TABLE 3.4: Capacités d'exploitation de l'intrus

3.3 Discussion

- Après l'étude de l'approche RSC, nous avons constaté que l'utilisation de deux clefs pour contrôler l'accès à un ambient rend l'approche plus complexe, alors que l'emploi d'une seule clef est suffisant.

- La notion de « *unprot* » utilisée dans cette approche et qui permet d'enlever la protection d'un ambient n'est pas nécessaire, car elle est similaire à la notion de «*prot*», et nous pouvons la définir via cette dernière (ex : $unprot_n^{k1} = prot_n^{\delta}$).

- L'approche de RSC peut être améliorée car elle ne simule pas la mobilité des ambients qu'on a besoin dans les réseaux AD HOC ; pour cela nous avons inspiré des capacités de mouvement de Cardelli et Gordon[40], et nous avons ajouté deux nouvelles capacités de mouvement d'ambient *in* et *out* ; la syntaxe de ces dernières est présentée ci-dessous.

- in_n^k : fait entrer l'ambient conteneur dans un ambient parallèle n en utilisant sa clef d'accès appropriée k.
- out_n^k : fait sortir l'ambient conteneur de son ambient parent n en utilisant sa clef d'accès appropriée k.

L'addition de ces deux nouvelles capacités nous a permis d'ajouter à l'approche RSC six autres règles de réduction présentées dans le tableau 3.6.

La règle (10) décrit le fait qu'un ambient peut entrer dans un ambient protégé. La capacité de sortir d'un ambient protégé est présentée par la règle (11). Les règles (12) et (13) dénotent le niveau de la protection offert par la clef publique δ, les tentatives d'un ambient pour accéder ou sortir d'un ambient protégé par δ sont réussies indépendamment de la clef utilisée. Le symbole $\ll \_ \gg$ désigne n'importe qu'elle valeur de la clef à employer par les capacités *in* et *out*.
Le calcul de [49, 50] permet d'identifier des menaces potentielles en combinant le $\oplus$, de in_n^k ou de out_n^k avec l'expression $(exp).\mathrm{P}$ des règles (14) et (15) dans le tableau 3.6 avec le choix compositionnel et le choix d'exécution de la relation de congruence structurelle présentée dans le tableau 3.2. Nous obtenons :

$$(in_n^k \oplus exp).\mathrm{P} \equiv (in_n^k.\mathrm{P}) \sqcap (exp.\mathrm{P}) \sqcap (in_n^k.exp.\mathrm{P}) \sqcap (exp.in_n^k.\mathrm{P}) \ldots\ldots(\mathrm{a})$$
$$(out_n^k \oplus exp).\mathrm{P} \equiv (out_n^k.\mathrm{P}) \sqcap (exp.\mathrm{P}) \sqcap (out_n^k.exp.\mathrm{P}) \sqcap (exp.out_n^k.\mathrm{P}) \ldots\ldots(\mathrm{b})$$

Par conséquent, l'intrus a un choix pour faire une des quatre options suivantes : utiliser la clef et cesser d'explorer, ignorer la clef et continuer l'exploration, utiliser la clef puis continuer d'explorer et, en conclusion, continuer d'explorer et utiliser la clef à un futur moment. Si on prend la relation (a), l'intrus choisit, par exemple, la troisième option $(in_n^k.exp.\mathrm{P})$, alors le résultat est un mouvement d'ambient à l'intérieur d'un autre ambient n. En plus, l'addition des deux capacités de mouvement à l'approche RSC (*in* et *out*) nous a permis d'ajouter quatre nouvelles capacités d'exploitation de l'intrus. Les scénarios illustrés dans e) et f) du tableau 3.7 définit comment un intrus gagne des capacités de mouvement d'ambient. Les scénarios g) et h) illustrent le fait que si la clef publique est employée pour la protection d'ambient, alors l'ambient contenant le processus d'intrus peut déplacer comme il veut.

Par conséquent, la nouvelle syntaxe et les nouvelles règles de réduction et capacités d'exploitation d'intrus de l'approche RSC sont présentées dans les tableaux 3.5, 3.6 et 3.7 respectivement :

n	$\in$	N	Nom
x	$\in$	V	Variable
k	$\in$	K$\cup\{\delta\}$	Clefs de sécurité
P,Q	::=		Processus
	|	0	Inactivité
	|	P | Q	Composition parallèle
	|	!P	Réplication
	|	${}^{k}_{n}$[a.P]	Séquence d'ambient
a,b	::=		Capacités de processus
	|	req_n^x	Demande de clefs
	|	pub_n^x	Publication de clefs
	|	mov_n^k	Mouvement d'un processus
	|	in_n^k	Mouvement d'entrée d'un ambient
	|	out_n^k	Mouvement de sortie d'un ambient
	|	exp	Exploration
	|	$prot_n^k$	Protection
	|	a $\oplus$ b	Choix compositionnel
	|	a $\sqcap$ b	Choix non-déterministe

Table 3.5: La nouvelle syntaxe de l'approche RSC

P' $\rightarrow$ Q' if P' $\equiv$ P,P $\rightarrow$ Q , Q $\equiv$ Q'	(1)
P $\mid$ R $\rightarrow$ Q $\mid$ R if P $\rightarrow$ Q	(2)
mov_n^k.P $\mid {}_n^k[\mathrm{Q}] \rightarrow {}_n^k[\mathrm{P} \mid \mathrm{Q}]$	(3)
${}_n^k[mov_n^k.\mathrm{P} \mid \mathrm{Q}] \rightarrow \mathrm{P} \mid {}_n^k[\mathrm{Q}]$	(4)
mov_n^-.P $\mid {}_n^\delta[\mathrm{Q}] \rightarrow {}_n^\delta[\mathrm{P} \mid \mathrm{Q}]$	(5)
${}_n^\delta[mov_n^-.\mathrm{P} \mid \mathrm{Q}] \rightarrow \mathrm{P} \mid {}_n^\delta[\mathrm{Q}]$	(6)
req_n^x.P $\mid (pub_n^k.\mathrm{Q}) \rightarrow \mathrm{P}[x \leftarrow k] \mid (pub_n^k.\mathrm{Q})$	(7)
$prot_n^k$.P $\mid {}_n^\delta[\mathrm{Q}] \rightarrow \mathrm{P} \mid {}_n^k[\mathrm{Q}]$	(8)
exp.P $\mid (pub_n^k.\mathrm{Q}) \rightarrow (mov_n^k \oplus exp).\mathrm{P} \mid (pub_n^k.\mathrm{Q})$	(9)
${}_n^\delta[in_m^k.\mathrm{P} \mid \mathrm{Q}] \mid {}_m^k[\mathrm{R}] \rightarrow {}_m^k[{}_n^\delta[\mathrm{P} \mid \mathrm{Q}]\mid \mathrm{R}]$	(10)
${}_m^k[{}_n^\delta[out_m^k.\mathrm{P} \mid \mathrm{Q}] \mid \mathrm{R}] \rightarrow {}_n^\delta[\mathrm{P} \mid \mathrm{Q}]\mid {}_m^k[\mathrm{R}]$	(11)
${}_m^\delta[in_n^-.\mathrm{P} \mid \mathrm{Q}] \mid {}_n^\delta[\mathrm{R}] \rightarrow {}_n^\delta[{}_m^\delta[\mathrm{P} \mid \mathrm{Q}] \mid \mathrm{R}]$	(12)
${}_m^\delta[{}_n^\delta[out_n^-.\mathrm{P} \mid \mathrm{Q}] \mid \mathrm{R}] \rightarrow {}_n^\delta[\mathrm{P} \mid \mathrm{Q}]\mid {}_m^\delta[\mathrm{R}]$	(13)
exp.P $\mid (pub_n^k.\mathrm{Q}) \rightarrow (in_n^k \oplus exp).\mathrm{P} \mid (pub_n^k.\mathrm{Q})$	(14)
exp.P $\mid (pub_n^k.\mathrm{Q}) \rightarrow (out_n^k \oplus exp).\mathrm{P} \mid (pub_n^k.\mathrm{Q})$	(15)

TABLE 3.6: Les nouvelles règles de réduction

a) exp.P $\mid (pub_n^k.\mathrm{Q}) \mid {}_n^k[\mathrm{R}] \rightarrow (pub_n^k.\mathrm{Q}) \mid {}_n^k[\mathrm{R} \mid exp.\mathrm{P}]$
b) ${}_n^k[exp.\mathrm{P} \mid (pub_n^k.\mathrm{Q}) \mid \mathrm{R}] \rightarrow exp.\mathrm{P} \mid {}_n^k[(pub_n^k.\mathrm{Q}) \mid \mathrm{R}]$
c) exp.P $\mid {}_n^\delta[\mathrm{R}] \rightarrow {}_n^\delta[\mathrm{R} \mid exp.\mathrm{P}]$
d) ${}_n^\delta[exp.\mathrm{P} \mid \mathrm{R}] \rightarrow exp.\mathrm{P} \mid {}_n^\delta[\mathrm{R}]$
e) ${}_m^{k'}[exp.\mathrm{P} \mid (pub_n^k.\mathrm{Q})] \mid {}_n^k[\mathrm{R}] \rightarrow {}_n^k\ [{}_m^{k'}[exp.\mathrm{P} \mid (pub_n^k.\mathrm{Q})]]$
f) ${}_n^k[{}_m^{k'}[exp.\mathrm{P} \mid (pub_n^k.\mathrm{Q})] \mid \mathrm{R}] \rightarrow {}_m^{k'}[exp.\mathrm{P} \mid (pub_n^k.\mathrm{Q})]\mid {}_n^k[\mathrm{R}]$
g) ${}_m^{k'}[exp.\mathrm{P}] \mid {}_n^\delta[\mathrm{R}] \rightarrow {}_n^\delta[{}_m^{k'}[exp.\mathrm{P}] \mid \mathrm{R}]$
h) ${}_n^\delta[{}_m^{k'}[exp.\mathrm{P}] \mid \mathrm{R}] \rightarrow {}_m^{k'}[exp.\mathrm{P}] \mid {}_n^\delta[\mathrm{R}]$

TABLE 3.7: Nouvelles capacités d'exploitation de l'intrus

3.4 Modélisation de l'attaque blackhole avec l'approche RSC

Notre exemple d'application, représenté dans la figure 3.1, est un système simple illustré par deux zones logiques : Entreprise et Réseau AD HOC appliquant un protocole de routage réactif et vulnérable à une attaque du trou noir (Blackhole). Pour rendre l'exemple plus facile à comprendre le nombre de machines est restreint. Le réseau AD HOC contient trois ordinateurs mobiles (a, b et c) ; la zone Entreprise contient le réseau AD HOC ainsi qu'un autre ordinateur présentant un intrus. Chaque ordinateur de l'entreprise est représenté par deux ambients imbriqués et protégés, l'ambient supérieur comporte : un processus représentant les ressources de l'ordinateur, un service de publication de clef d'accès vers le sous ambient associé qui contient un processus représentant les données de la table de routage, ce sous-ambient est nommé *tr* suivi du nom de l'ordinateur approprié(même nom est affecté à son processus mais en majuscule). Les noms d'ambients et de processus des ordinateurs de l'entreprise sont basés sur la première lettre du nom de l'ordinateur. Par exemple, l'ambient représentant l' « Ordinateur a » est nommé *a* et contient le processus A, et ainsi de suite.

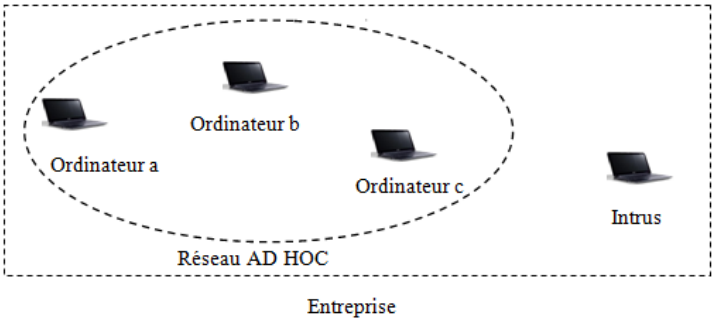

FIGURE 3.1: Réseau AD HOC vulnérable à l'attaque du blackhole

Le réseau AD HOC est illustré par un processus « R » fonctionnant dans un ambient protégé. Dans le nouveau calcul de [49, 50], chaque ambient est protégé par une clef d'accès. Une clef peut avoir la valeur publique par défaut δ ou une valeur différente secrète imposée par la politique de sécurité. La valeur δ assignée aux clefs déclare qu'il n'y a pas de restriction pour l'action d'entrée ou de sortie. La hiérarchie des ambients

et des sous-ambients reflète la topologie du réseau représentée sur la figure 3.1. Ainsi, l'ambient r est un ambient de niveau supérieur. Nous spécifions trois sous ambients dans l'ambient r : a, b et c. L'intrus une fois entrer dans le réseau AD HOC peut choisir ces cibles arbitrairement, dans la suite de l'exemple nous supposons que l'intrus choisit les cibles « ordinateur a » et « ordinateur c », donc, à un moment donné ce nœud malveillant peut recevoir un ROUTE_REQUEST de la part de l' « Ordinateur a » vers la destination « Ordinateur c » et ensuite attaquer le protocole de routage (attaque du blackhole) en renvoyant à l' « Ordinateur a » un ROUTE_REPLY qui contient des informations falsifiées spécifiant le chemin optimal vers la destination.
Le système qui représente le réseau AD HOC est indiqué par le processus suivant :
$S = {}^{k1}_{r}\ [R|\ {}^{\delta}_{a}\ [A|\ !(pub^{k2}_{tra}.0)|\ {}^{k2}_{tra}[TRA]]\ |\ {}^{\delta}_{b}\ [B|\ !(pub^{k3}_{trb}.0)|\ {}^{k3}_{trb}[TRB]]\ |\ {}^{\delta}_{c}\ [C|\ !(pub^{k4}_{trc}.0)|\ {}^{k4}_{trc}[TRC]]]$

3.4.1 Processus d'intrus

Le processus d'intrus représenté par trois processus qui s'exécutent en parallèle : un service de publication vers la table de routage, le processus de la table de routage qui s'exécute à l'intérieur d'un ambient protégé et un autre processus spécifiant l'envoi du message ROUTE_REPLY dont le but est d'effectuer une attaque du blackhole. L'intrus se connecte au réseau AD HOC pour explorer le système. L'opérateur *exp* modélise la manière dont l'intrus explore le système et gagne des capacités en termes d'action *mov*, *in* ou *out*. Cet opérateur est le constructeur syntaxique qui imite le comportement non déterministe de l'intrus et il donne à l'intrus le choix d'employer ces clefs et de lancer son exploration.

Le processus d'intrus est modélisé comme suit :

$$I = in^{k1}_{r} \oplus mov^{\delta}_{i} \oplus mov^{\delta}_{a} \oplus exp.I'|\ !(pub^{k5}_{tri}.0)|\ {}^{k5}_{tri}[TRI]$$

Il y a soixante-quatre comportements possibles de l'intrus présentés par les trois opérateurs $\oplus$ dans l'expression ci-dessus. Considérant le cas simplifié de a, b, c et d, les choix sont :

$a \oplus b \oplus c \oplus d = a \sqcap b \sqcap c \sqcap d \sqcap a.b \sqcap a.c \sqcap a.d \sqcap b.a \sqcap b.c \sqcap b.d \sqcap c.a \sqcap c.b \sqcap c.d \sqcap d.a \sqcap d.b \sqcap d.c \sqcap a.b.c \sqcap b.a.c \sqcap a.c.b \sqcap b.c.a \sqcap c.a.b \sqcap c.b.a \sqcap b.c.d \sqcap c.b.d \sqcap b.d.c \sqcap c.d.b \sqcap d.b.c \sqcap d.c.b \sqcap a.b.d \sqcap b.a.d \sqcap a.d.b \sqcap b.d.a \sqcap d.a.b \sqcap d.b.a \sqcap c.d.a \sqcap d.c.a \sqcap c.a.d \sqcap d.a.c \sqcap a.c.d \sqcap a.d.c \sqcap a.b.c.d \sqcap a.b.d.c \sqcap a.c.b.d \sqcap a.c.d.b \sqcap a.d.b.c \sqcap a.d.c.b \sqcap b.a.c.d \sqcap b.a.d.c \sqcap b.c.a.d \sqcap b.c.d.a \sqcap b.d.a.c \sqcap b.d.c.a \sqcap c.a.b.d \sqcap c.a.d.b \sqcap c.b.a.d \sqcap c.b.d.a \sqcap c.d.a.b \sqcap c.d.b.a \sqcap d.a.b.c \sqcap d.a.c.b \sqcap d.b.a.c \sqcap d.b.c.a \sqcap d.c.a.b \sqcap d.c.b.a.$

Le meilleur comportement que l'intrus peut choisir est :

I= $in_r^{k1}.mov_i^{\delta}.mov_a^{\delta}.exp$.I'| !($pub_{tri}^{k5}.0$)| $_{tri}^{k5}$[TRI]

Les autres comportements sont ignorés pour les raisons suivantes : ceux qui ne produisent aucun résultat ou ils paralysent l'attaque. L'intrus entre dans l'ambient r pour se connecter au réseau AD HOC puisqu'il connait au commencement la clef d'accès à cet ambient ; ensuite, le processus d'intrus sort de l'ambient i par l'exécution de mov_i^{δ} pour entrer dans l'ambient a en exécutant mov_a^{δ}, puis lance l'exploration. Le tableau 3.8 montre les étapes de l'exploration .

$$
\begin{array}{lll}
\mathrm{S}|_i^{\delta}[\ \mathrm{I}\ ] & \rightarrow & {}_r^{k1}[\mathrm{R}|_a^{\delta}[\mathrm{A}|\ !(pub_{tra}^{k2}.0)|_{tra}^{k2}[\mathrm{TRA}]]|_b^{\delta}[\mathrm{B}|\ !(pub_{trb}^{k3}.0)|_{trb}^{k3}[\mathrm{TRB}]]|_c^{\delta}[\mathrm{C}|\ !(pub_{trc}^{k4}.0)| \\
 & & {}_{trc}^{k4}[\mathrm{TRC}]]]\ |_i^{\delta}[in_r^{k1}.mov_i^{\delta}.mov_a^{\delta}.exp.\mathrm{I'}|\ !(pub_{tri}^{k5}.0)|\ {}_{tri}^{k5}[\mathrm{TRI}]] \\
 & \rightarrow & {}_r^{k1}[\mathrm{R}|_i^{\delta}[mov_i^{\delta}.mov_a^{\delta}.exp.\mathrm{I'}|\ !(pub_{tri}^{k5}.0)|_{tri}^{k5}[\mathrm{TRI}]]|_a^{\delta}[\mathrm{A}|\ !(pub_{tra}^{k2}.0)|_{tra}^{k2}[\mathrm{TRA}]]| \\
 & & {}_b^{\delta}[\mathrm{B}|\ !(pub_{trb}^{k3}.0)|_{trb}^{k3}[\mathrm{TRB}]]|_c^{\delta}[\mathrm{C}|\ !(pub_{trc}^{k4}.0)|_{trc}^{k4}[\mathrm{TRC}]]] \\
 & \rightarrow & {}_r^{k1}[\mathrm{R}|_i^{\delta}[\ !(pub_{tri}^{k5}.0)|_{tri}^{k5}[\mathrm{TRI}]]|\quad mov_a^{\delta}.exp.\mathrm{I'}|_a^{\delta}[\mathrm{A}|\ !(pub_{tra}^{k2}.0)|_{tra}^{k2}[\mathrm{TRA}]]|_b^{\delta}[\mathrm{B}| \\
 & & !(pub_{trb}^{k3}.0)|_{trb}^{k3}[\mathrm{TRB}]]|_c^{\delta}[\mathrm{C}|\ !(pub_{trc}^{k4}.0)|_{trc}^{k4}[\mathrm{TRC}]]] \\
 & \rightarrow & {}_r^{k1}[\mathrm{R}|_i^{\delta}[\ !(pub_{tri}^{k5}.0)|_{tri}^{k5}[\mathrm{TRI}]]|_a^{\delta}[exp.\mathrm{I'}|\mathrm{A}|\ !(pub_{tra}^{k2}.0)|_{tra}^{k2}[\mathrm{TRA}]]| \\
 & & {}_b^{\delta}[\mathrm{B}|\ !(pub_{trb}^{k3}.0)|_{trb}^{k3}[\mathrm{TRB}]]|_c^{\delta}[\mathrm{C}|\ !(pub_{trc}^{k4}.0)|_{trc}^{k4}[\mathrm{TRC}]]] \\
 & \rightarrow & {}_r^{k1}[\mathrm{R}|_i^{\delta}[\ !(pub_{tri}^{k5}.0)|_{tri}^{k5}[\mathrm{TRI}]]|_a^{\delta}[mov_{tra}^{k2}\ \oplus\ exp.\mathrm{I'}|\mathrm{A}|\ !(pub_{tra}^{k2}.0)|_{tra}^{k2}[\mathrm{TRA}]]| \\
 & & {}_b^{\delta}[\mathrm{B}|\ !(pub_{trb}^{k3}.0)|_{trb}^{k3}[\mathrm{TRB}]]|_c^{\delta}[\mathrm{C}|\ !(pub_{trc}^{k4}.0)|_{trc}^{k4}[\mathrm{TRC}]]] \\
 & \rightarrow & {}_r^{k1}[\mathrm{R}|_i^{\delta}[\ !(pub_{tri}^{k5}.0)|_{tri}^{k5}[\mathrm{TRI}]]|_a^{\delta}[mov_{tra}^{k2}.exp.\mathrm{I'}|\mathrm{A}|\ !(pub_{tra}^{k2}.0)|_{tra}^{k2}[\mathrm{TRA}]]| \\
 & & {}_b^{\delta}[\mathrm{B}|\ !(pub_{trb}^{k3}.0)|_{trb}^{k3}[\mathrm{TRB}]]|_c^{\delta}[\mathrm{C}|\ !(pub_{trc}^{k4}.0)|_{trc}^{k4}[\mathrm{TRC}]]] \\
 & \rightarrow & {}_r^{k1}[\mathrm{R}|_i^{\delta}[\ !(pub_{tri}^{k5}.0)|_{tri}^{k5}[\mathrm{TRI}]]|_a^{\delta}[\mathrm{A}|\ !(pub_{tra}^{k2}.0)|_{tra}^{k2}[exp.\mathrm{I'}|\mathrm{TRA}]]| \\
 & & {}_b^{\delta}[\mathrm{B}|\ !(pub_{trb}^{k3}.0)|_{trb}^{k3}[\mathrm{TRB}]]|_c^{\delta}[\mathrm{C}|\ !(pub_{trc}^{k4}.0)|_{trc}^{k4}[\mathrm{TRC}]]]
\end{array}
$$

Table 3.8: Comportement du processus d'intrus

3.4.2 Sécurisation du système

Après la modélisation de l'attaque blackhole sur un réseau AD HOC avec la nouvelle approche de RSC, nous avons constaté que la requête ROUTE_REQUEST émise par l' « ordinateur a » sera reçue par la machine de l'intrus, en suite ce dernier émet sa réponse falsifiée ROUTE_REPLY afin d'intercepter d'une manière transparente tous les paquets échangés entre l' « ordinateur a » et l' « ordinateur c » et par la suite effectuer un déni de service (DoS). Le problème qui surgit est, comment modifier le système initial et le transformer en un qui soit sécurisé ?

Plusieurs approches et solutions ont été proposées pour sécuriser le routage dans les réseaux mobiles AD HOC, parmi celles-ci l'utilisation d'un IDS distributif et coopératif proposée par Zhang, Lee et Huang[51, 52], la vérification de l'authenticité des nœuds en utilisant la redondance du réseau ou l'utilisation des numéros de séquences pour les paquets proposées par Mohammad Al-Shurman, Seong-Moo Yoo et Seungjin Park dans [53].

Nous avons exploité une solution qui consiste à protéger chaque ordinateur avec une clef privée, de plus, pour s'assurer qu'une ancienne clef, possiblement compromise, n'est pas utilisée par un intrus, les clefs des ordinateurs a, b et c seront mis-à-jour (changement de clef) quant un nouveau système va joindre le réseau AD HOC.

Le renforcement du système se réalise grâce à l'utilisation de l'action *prot* par chaque ordinateur. Le tableau 3.9 montre les étapes de renforcement du système.

S'=	$\overset{k1}{r}[prot_a^{ka}.prot_b^{kb}.prot_c^{kc}.\mathrm{R} \quad \|_a^{\delta}[\mathrm{A}\|\,!(pub_{tra}^{k2}.0)\|_{tra}^{k2}[\mathrm{TRA}]]\|_b^{\delta}[\mathrm{B}\|\,!(pub_{trb}^{k3}.0)\|_{trb}^{k3}[\mathrm{TRB}]]\|$ $\overset{\delta}{c}[\mathrm{C}\|\,!(pub_{trc}^{k4}.0)\|_{trc}^{k4}[\mathrm{TRC}]]]$
$\rightarrow$	$\overset{k1}{r}[prot_b^{kb}.prot_c^{kc}.\mathrm{R} \quad \|_a^{ka}[\mathrm{A}\|\,!(pub_{tra}^{k2}.0)\|_{tra}^{k2}[\mathrm{TRA}]]\|_b^{\delta}[\mathrm{B}\|\,!(pub_{trb}^{k3}.0)\|_{trb}^{k3}[\mathrm{TRB}]]\|_c^{\delta}[\mathrm{C}\|$ $!(pub_{trc}^{k4}.0)\|_{trc}^{k4}[\mathrm{TRC}]]]$
$\rightarrow$	$\overset{k1}{r}[prot_c^{kc}.\mathrm{R}\,\|_a^{ka}[\mathrm{A}\|\,!(pub_{tra}^{k2}.0)\|_{tra}^{k2}[\mathrm{TRA}]]\|_b^{kb}[\mathrm{B}\|\,!(pub_{trb}^{k3}.0)\|_{trb}^{k3}[\mathrm{TRB}]]\|_c^{\delta}[\mathrm{C}\|\,!(pub_{trc}^{k4}.0)\|$ $\overset{k4}{trc}[\mathrm{TRC}]]]$
$\rightarrow$	$\overset{k1}{r}[\mathrm{R}\|_a^{ka}[\mathrm{A}\|\,!(pub_{tra}^{k2}.0)\|_{tra}^{k2}[\mathrm{TRA}]]\|_b^{kb}[\mathrm{B}\|\,!(pub_{trb}^{k3}.0)\|_{trb}^{k3}[\mathrm{TRB}]]\|_c^{kc}[\mathrm{C}\|\,!(pub_{trc}^{k4}.0)\|$ $\overset{k4}{trc}[\mathrm{TRC}]]]$

TABLE 3.9: Étapes de renforcement du système

Le processus d'intrus dans le nouveau système S' est modélisé comme suite :
$\mathrm{I} = in_r^{k1} \oplus mov_i^{\delta} \oplus exp.\mathrm{I'}|\,!(pub_{tri}^{k5}.0)|\ \overset{k5}{tri}[\mathrm{TRI}]]$
Il y a quinze comportements possibles de l'intrus présentés par les deux opérateurs $\oplus$ dans l'expression ci-dessus. Considérant le cas simplifié de a, b et c, les choix sont :
$a \oplus b \oplus c = a \sqcap b \sqcap c \sqcap a.b \sqcap a.c \sqcap b.a \sqcap b.c \sqcap c.a \sqcap c.b \sqcap a.b.c \sqcap b.a.c \sqcap a.c.b \sqcap b.c.a \sqcap c.a.b \sqcap c.b.a.$
Le meilleur comportement que l'intrus peut choisir est :
$\mathrm{I} = in_r^{k1}.mov_i^{\delta}.exp.\mathrm{I'}|\,!(pub_{tri}^{k5}.0)|\ \overset{k5}{tri}[\mathrm{TRI}]]$

Les autres comportements sont ignorés pour les raisons suivantes : ceux qui ne produisent aucun résultat ou ils paralysent l'attaque. L'exploration de l'intrus dans le nouveau système s' est similaire à celle du système S, lorsqu'il sort de l'ambient i il sera bloqué et n'entre pas dans l'ambient a, puisqu'il ne connait pas la clef d'accès vers cet ambient (ka). Les étapes de l'exploration de l'intrus sont présentées dans le tableau 3.10

$$
\begin{array}{lll}
\text{S'}|_i^{\delta}[\,\text{I}\,] & \rightarrow & {}^{k1}_{r}[\text{R}|^{ka}_{a}[\text{A}|\,!(pub^{k2}_{tra}.0)|^{k2}_{tra}[\text{TRA}]]|^{kb}_{b}[\text{B}|\,!(pub^{k3}_{trb}.0)|^{k3}_{trb}[\text{TRB}]]|^{kc}_{c}[\text{C}|\,!(pub^{k4}_{trc}.0)| \\
& & {}^{k4}_{trc}[\text{TRC}]]]\ |^{\delta}_{i}[in^{k1}_{r}.mov^{\delta}_{i}.exp.\text{I'}|\,!(pub^{k5}_{tri}.0)|\ {}^{k5}_{tri}[\text{TRI}]] \\
& \rightarrow & {}^{k1}_{r}[\text{R}|^{\delta}_{i}[mov^{\delta}_{i}.exp.\text{I'}|\,!(pub^{k5}_{tri}.0)|^{k5}_{tri}[\text{TRI}]]|^{ka}_{a}[\text{A}|\,!(pub^{k2}_{tra}.0)|^{k2}_{tra}[\text{TRA}]]| \\
& & {}^{kb}_{b}[\text{B}|\,!(pub^{k3}_{trb}.0)|^{k3}_{trb}[\text{TRB}]]|^{kc}_{c}[\text{C}|\,!(pub^{k4}_{trc}.0)|^{k4}_{trc}[\text{TRC}]]] \\
& \rightarrow & {}^{k1}_{r}[\text{R}|^{\delta}_{i}[\,!(pub^{k5}_{tri}.0)|^{k5}_{tri}[\text{TRI}]]|\ exp.\text{I'}|^{ka}_{a}[\text{A}|\,!(pub^{k2}_{tra}.0)|^{k2}_{tra}[\text{TRA}]]|^{kb}_{b}[\text{B}|\,!(pub^{k3}_{trb}.0)| \\
& & {}^{k3}_{trb}[\text{TRB}]]|^{kc}_{c}[\text{C}|\,!(pub^{k4}_{trc}.0)|^{k4}_{trc}[\text{TRC}]]]
\end{array}
$$

TABLE 3.10: Exploration de l'intrus dans le système renforcé

3.5 Conclusion

Dans ce chapitre nous avons appliqué l'approche améliorée de RSC afin de spécifier un réseau AD HOC, et elle nous a permis de montrer que notre réseau n'est pas sécurisé et vulnérable à une attaque du blackhole. Pour renforcer la sécurité de notre système et prévenir à cette attaque, nous avons proposé une autre spécification du réseau représentée par une protection avec des clefs privées.

Dans le chapitre suivant nous allons décrire en détaille le fonctionnement de l'application. Cette dernière consiste en une spécification des réseaux informatiques.

Chapitre 4

Implémentation

4.1 Introduction

Dans ce chapitre, nous allons décrire en détail le fonctionnement de l'application de génération de spécification. Cette dernière est réalisée par Marc Saint-Laurent[54] dont le but est la spécification de propriétés de systèmes informatiques. Nous avons implémenté les règles de réduction sous cette application afin d'appliquer l'approche RSC sur notre système de spécification présenté dans le chapitre 3.

Dans la première partie, nous allons présenter le langage de programmation utilisé, ainsi que son environnement et ces outils nécessaires. Par la suite, nous allons présenter les règles de réduction, puis, nous expliquerons le déroulement de notre système de spécification.

4.2 Langage de programmation

Marc Saint-Laurent[54] a utilisé JAVA pour développer l'application de « genspec », en effet, Java est un langage de programmation objet dont les premières versions datent de 1995 et il a été mis au point par la firme Sun Microsystems. Le langage Java a une syntaxe très proche du langage C++ et répond aux trois principes fondamentaux de l'approche orientée objet (POO) : l'encapsulation, le polymorphisme et l'héritage. Ce langage est caractérisé par sa facilité d'utilisation.

4.2.1 Environnement et outils de développement

Marc Saint-Laurent a utilisé dans son application les outils suivants :

- **JDK (Java Development Kit)** :

  L'environnement dans lequel le code java est compilé pour être transformé en

ByteCode afin que la JVM (Java Virtuel Machine) de java puisse l'interpréter.

- **L'environnement NetBeans IDE** :

  NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2 (Common Development and Distribution License).

  L'EDI NetBeans est un outil conçue pour écrire, compiler, déboguer et déployer des programmes. Il est écrit en Java mais peut supporter n'importe quel langage de programmation[11].

- **L'environnement Eclipse IDE** :

  Eclipse IDE est un environnement de développement intégré libre, extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java à l'aide de la bibliothèque graphique SWT d'IBM et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions.

  La spécificité d'Eclipse IDE vient du fait de son architecture totalement développée autour de la notion de plugin (en conformité avec la norme OSGi) : toutes les fonctionnalités de cet atelier sont développées en tant que plug-in[55].

Cette application est développée principalement avec l'environnement de développement Eclipse, version 3.4.2. NetBeans est utilisé juste pour créer quelques interfaces graphiques.

4.3 Structure de l'application

La fenêtre principale (figure 4.1) se compose d'une barre de menus, une barre d'outils, le reste de la fenêtre est occupé par un espace de travail divisé en deux sections :

- La première section qui représente graphiquement la topologie du réseau.
- La deuxième section affiche la spécification du réseau.

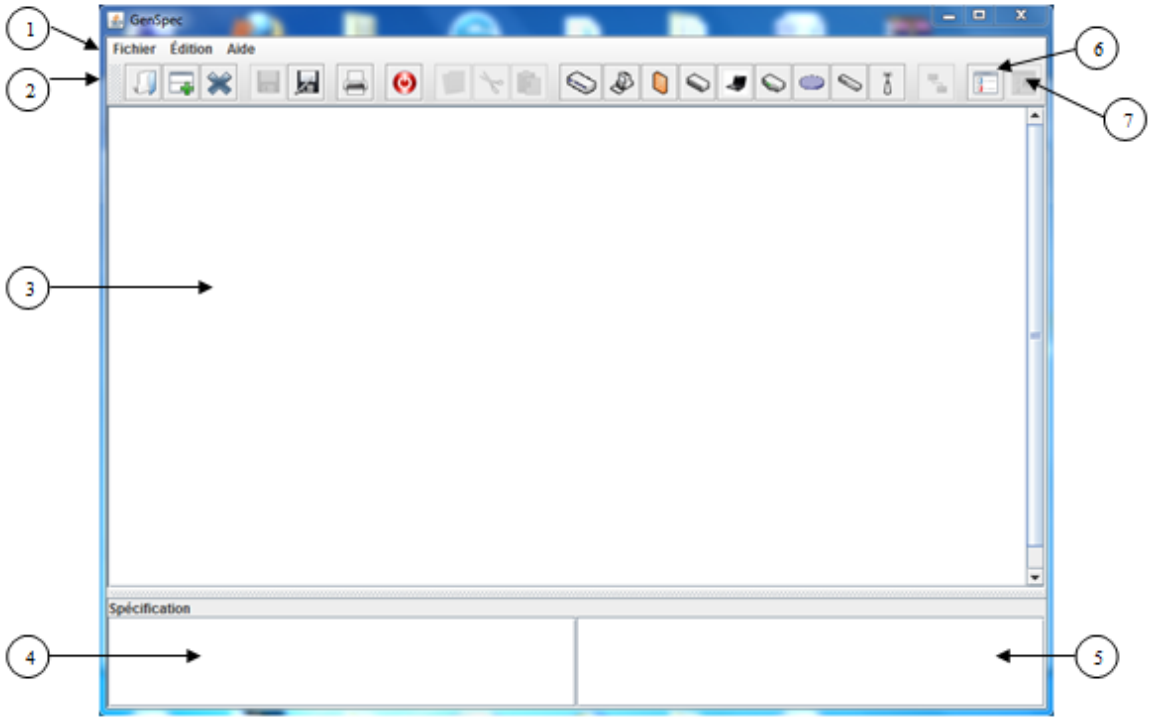

FIGURE 4.1: La fenêtre principale.

(1) : La barre de menu qui permet d'accéder aux fonctions de l'application. (figure 4.1)

(2) : La barre d'outils regroupe plusieurs boutons, et des icônes qui peuvent être retirées ou ajoutées de l'interface graphique.(figure 4.1)

(3) : Le composant qui représente graphiquement la topologie du réseau. (figure 4.1)

(4) : Le composant qui affiche la spécification de la topologie dans son état actuel. (figure 4.1)

(5) : Le composant qui permet d'obtenir la spécification en mode texte. (figure 4.1)

(6) : Le bouton qui permet de voir et modifier la spécification. (figure 4.2 et figure 4.3)

(7) : Le bouton qui permet de lancer la réduction. (figure 4.4)

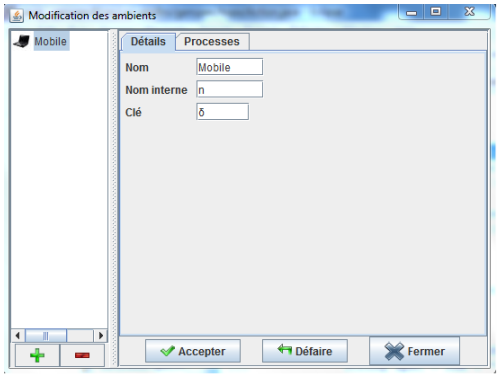

FIGURE 4.2: Voir les détails(Détails).

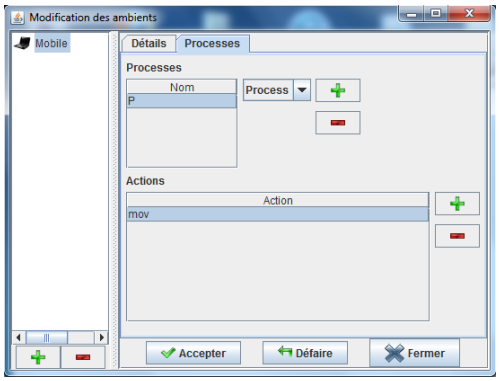

FIGURE 4.3: Détails(Processes).

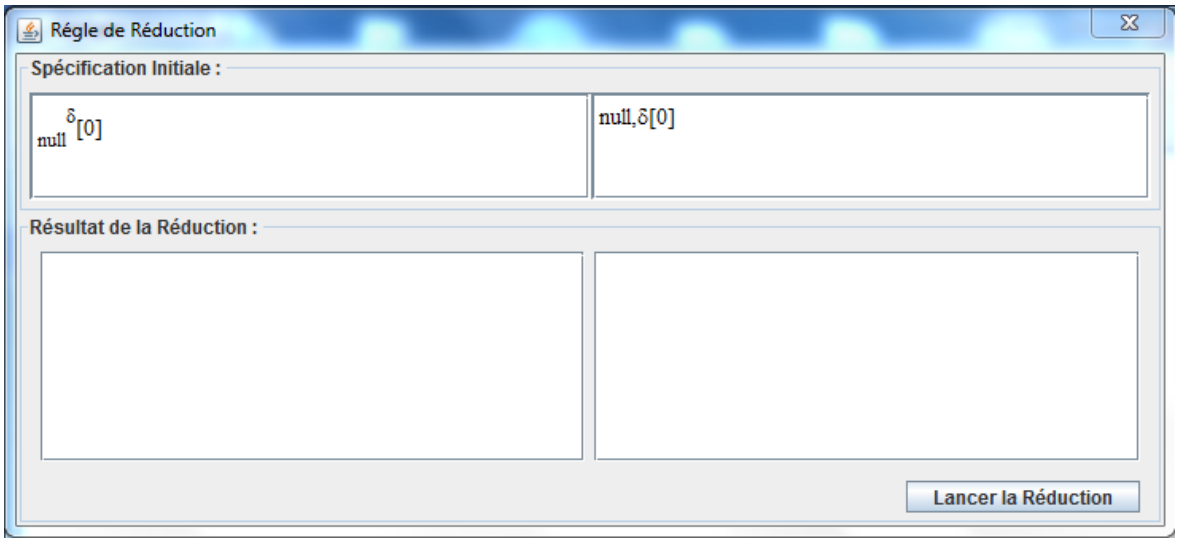

FIGURE 4.4: La fenêtre de la réduction.

4.4 Déroulement de l'application

En premier lieu, il faut d'abord introduire une topologie, ensuite définir les paramètres des nœuds, cliquer sur le bouton « Règles de réduction » et enfin pour voir le résultat de réduction il faut cliquer sur le bouton « lancer la réduction ».

4.4.1 Résultat de l'exécution

4.4.1.1 Les règles de réduction

a. La règle numéro 3 :

Dans la figure 4.5, nous présentons le résultat de l'exécution de la règle numéro 3, celle-ci décrit le fait qu'un processus peut entrer dans un ambient protégé.

b. La règle numéro 4 :

Dans la figure 4.6, nous présentons le résultat de l'exécution de la règle numéro 4, celle-ci décrit la capacité de sortir d'un ambient protégé.

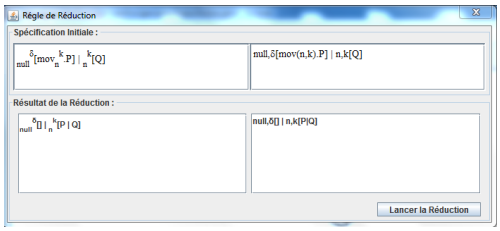

FIGURE 4.5: Résultat de la règle 3.

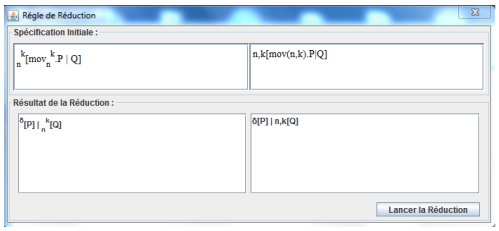

FIGURE 4.6: Résultat de la règle 4.

c. La règle numéro 5 :

Dans la figure 4.7, nous présentons le résultat de l'exécution de la règle numéro 5, celle-ci décrit le fait qu'un processus peut entrer dans un ambient protégé avec une clef publique en utilisant n'importe quelle clef.

d. La règle numéro 6 :

Dans la figure 4.8, nous présentons le résultat de l'exécution de la règle numéro 6, celle-ci décrit la capacité de sortir d'un ambient protégé avec une clef publique en utilisant n'importe quelle clef.

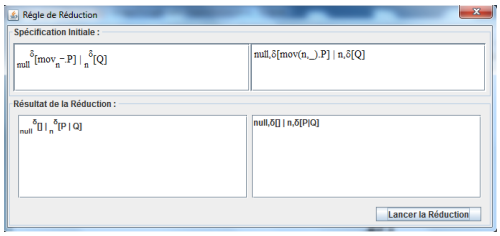

FIGURE 4.7: Résultat de la règle 5.

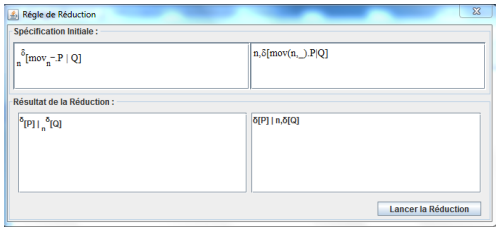

FIGURE 4.8: Résultat de la règle 6.

e. La règle numéro 7 :

Dans la figure 4.9, nous présentons le résultat de l'exécution de la règle numéro 7, celle-ci décrit une demande de la clef appropriée du service de publication approprié.

f. La règle numéro 8 :

Dans la figure 4.10, nous présentons le résultat de l'exécution de la règle numéro 8, celle-ci décrit le renforcement des clefs de sécurité.

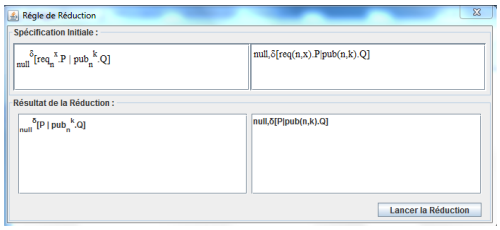

FIGURE 4.9: Résultat de la règle 7.

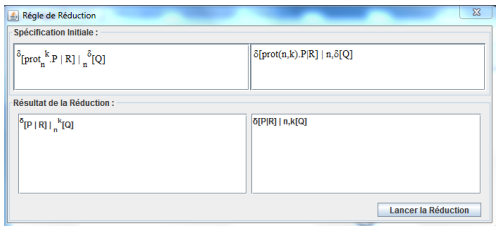

FIGURE 4.10: Résultat de la règle 8.

g. Les règles numéro 9, 14, et 15 :

Dans la figure 4.11, nous présentons le résultat de l'exécution des règles numéro 9,14 et 15, celles-ci décrivent le comportement de l'intrus. Elles modélisent la manière dont le processus d'intrus peut employer un mouvement d'ambient ou de processus, d'accès ou de sortie s'il aboutie à n'importe quel privilège pour un certain scénario d'attaque. Cette relation s'ajoute à la connaissance de l'intrus et lui donne de nouvelles capacités de mouvement.

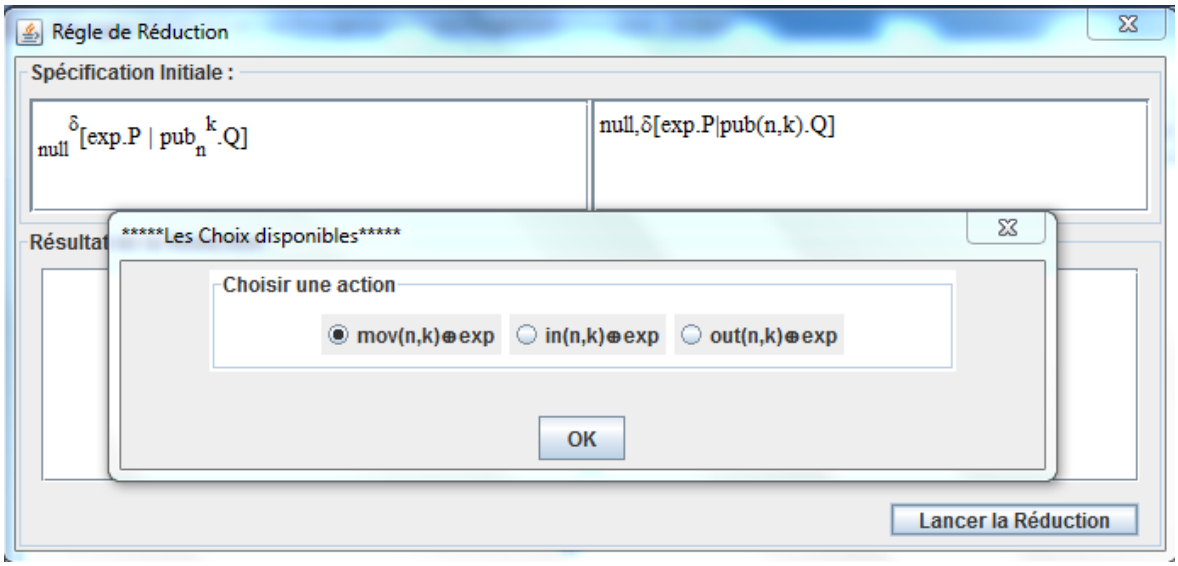

FIGURE 4.11: Résultat des règles 9,14,et 15.

h. La règle numéro 10 :

Dans la figure 4.12, nous présentons le résultat de l'exécution de la règle numéro 10, celle-ci décrit le fait qu'un ambient peut entrer dans un ambient protégé.

i. La règle numéro 11 :

Dans la figure 4.13, nous présentons le résultat de l'exécution de la règle numéro 11, celle-ci décrit la capacité d'un ambient de sortir d'un autre ambient protégé.

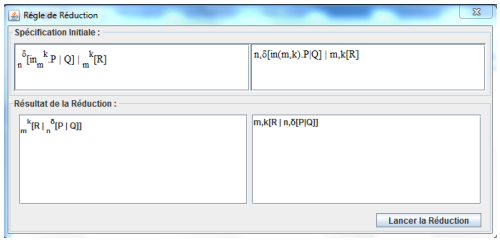

FIGURE 4.12: Résultat de la règle 10.

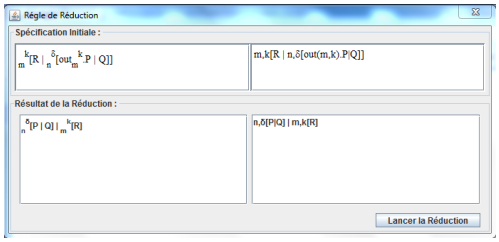

FIGURE 4.13: Résultat de la règle 11.

j. La règle numéro 12 :

Dans la figure 4.14, nous présentons le résultat de l'exécution de la règle numéro 12, celle-ci décrit le fait qu'un ambient peut entrer dans un ambient protégé avec une clef publique en utilisant n'importe qu'elle clef.

k. La règle numéro 13 :

Dans la figure 4.15, nous présentons le résultat de l'exécution de la règle numéro 13, celle-ci décrit la capacité d'un ambient de sortir d'un autre ambient protégé avec une clef publique en utilisant n'importe quelle clef.

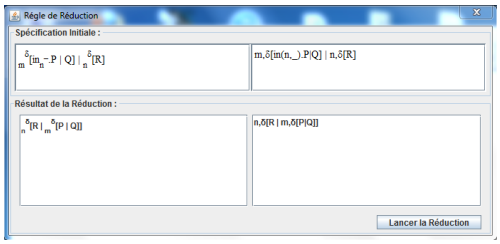

FIGURE 4.14: Résultat de la règle 12.

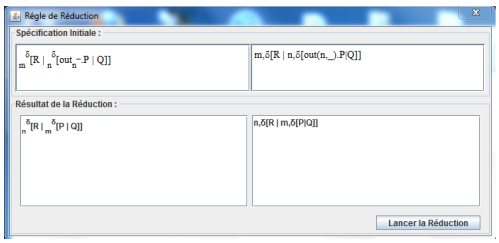

FIGURE 4.15: Résultat de la règle 13.

4.4.1.2 Spécification de l'attaque blackhole

La figure 4.16 montre la topologie d'un réseau vulnérable à l'attaque blackhole présentée dans le chapitre 3, et la figure 4.17 présente le résultat finale de l'exploration

de l'intrus. La figure 4.18 montre le résultat finale de l'exploration de l'intrus après la sécurisation du système.

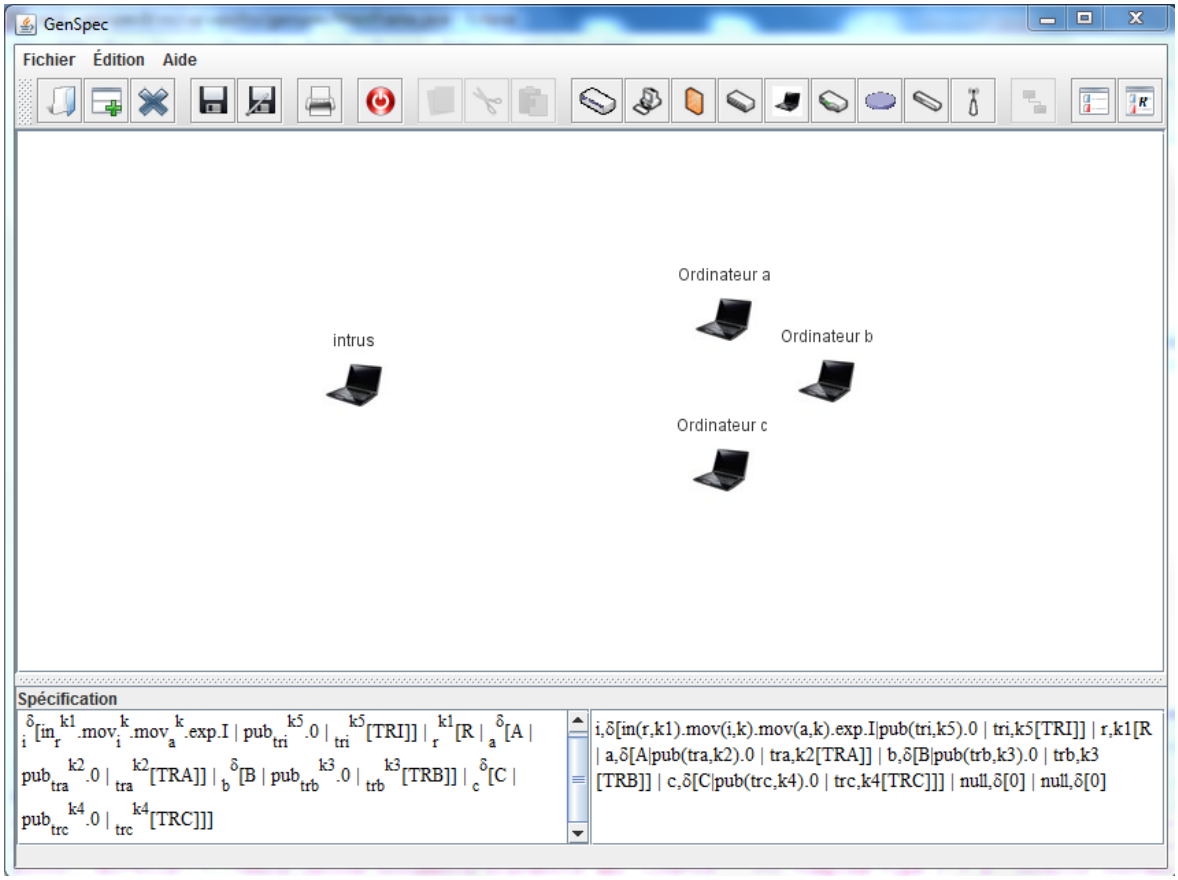

FIGURE 4.16: Spécification de l'attaque blackhole.

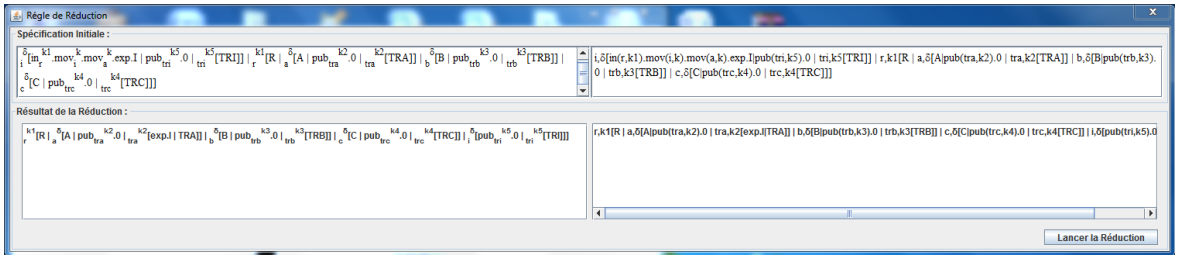

FIGURE 4.17: Résultat de l'exploration de l'intrus.

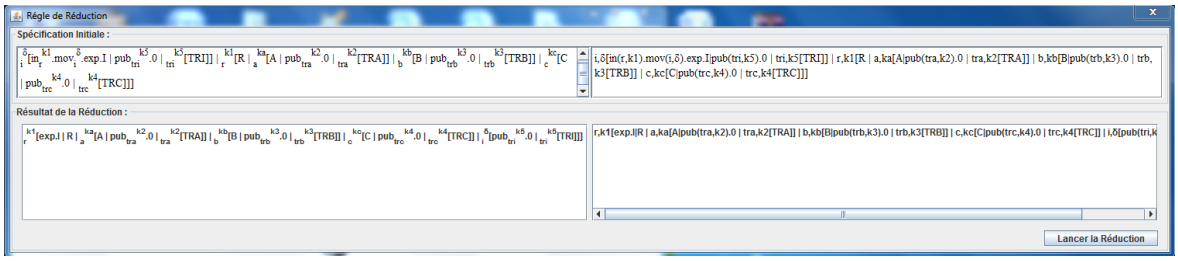

FIGURE 4.18: Résultat de l'exploration de l'intrus après la sécurisation du système.

4.5 Conclusion

Dans ce chapitre, nous avons montré toutes les règles de réduction de l'approche RSC implémentées sur l'application de Marc saint-Laurent, ainsi que la modélisation de l'attaque blackhole sur un réseau AD HOC. Pour prouver que l'application est fonctionnelle sur plusieurs exemples, nous avons rajouté un annexe comportant la modélisation de deux autres systèmes informatiques.

Conclusion Générale et Perspectives

Dans ce mémoire, nous avons présenté, dans le premier chapitre, une vue globale sur la sécurité informatique en décrivant les menaces et les risques, la notion de politique de sécurité, en plus, nous avons présenté l'architecture de la sécurité informatique et quelques techniques de défense, ainsi qu'un aperçu sur la sécurité dans les réseaux AD HOC.

Dans le deuxième chapitre, nous avons rappelé certaines algèbres de processus. En premier lieu, nous avons décrit le CCS qui est un langage mathématique permettant de décrire les systèmes concurrents. Ensuite, nous avons décrit le CSP qui permet de modéliser l'interaction des systèmes. Puis nous nous sommes intéressé aux algèbres dérivés du CCS, qui sont le π-calcul et le calcul ambient, ce dernier offre la possibilité d'exprimer tous les aspects de la mobilité. Nous avons passé en revue quelques travaux de recherche récents dans le domaine de la recherche de spécification des systèmes informatiques.

Dans le troisième chapitre, nous avons décrit la syntaxe et la sémantique de l'approche RSC basée sur le calcul ambient, ensuite nous avons présenté les améliorations apportées à cette approche afin de l'appliquée aux réseaux AD HOC. Et pour conclure nous avons montré un cas d'étude qui consiste à l'application de l'approche RSC sur un réseau AD HOC vulnérable à l'attaque du trou noir (blackhole). Dans le quatrième chapitre, nous avons présenté l'implémentation des règles de réduction de l'approche RSC sur l'application de Marc Saint-Laurent, ainsi les résultats de l'exploration de

l'intrus dans les deux systèmes, l'initial et le renforcé présentés dans le troisième chapitre.

Comme perspectives, nous envisageons d'appliquer l'approche RSC sur les réseaux de capteurs, d'améliorer l'application pour la rendre plus performante et de spécifier d'autres systèmes plus complexes que celui modélisé dans ce mémoire.

Bibliographie

[1] Télécom Bretagne. *"Introduction à la sécurité informatique."* École pionnière en formation, en recherche et en entrepreneuriat, campus de Toulouse ; Cours : sécurité des systèmes informatiques.

[2] http ://xenod.free.fr/0_La_securite_informatique.html.Le 13 février 2013.

[3] http ://www.authsecu.com/internet-et-la-vie-prive/internet-et-la-vie-prive.php. Le 13 février 2013.

[4] http ://www.securiteinfo.com. Le 17 février 2013.

[5] Wikimedia Foundation, Inc. "Politique de sécurité des réseaux informatiques". Le 17 novembre 2009.

[6] Laurent BLOCH, Christophe WOLFHUGEL, "Sécurité informatique principe et méthode". 2éme édition Eyrolles Paris 2009.

[7] Jérôme ATHIAS Consultant sécurité des systèmes d'information, "La politique de sécurité informatique". France Juillet 2002.

[8] Arnaud CHARLIER,"Introduction à la sécurité informatique". Paris academy of computer science of Supinfo projects, 30/05/2006 Supinfo Paris.

[9] http ://www.securiteinfo.com/attaques/hacking/typesattaques.html

[10] http ://www.z0rglub.com/piratag.

[11] http ://fr.wikipedia.org/wiki.

[12] E. Cole. *"Hackers Attention, catégorie : systèmes et réseaux/sécurité configuration."* Edition campus press, 2009.

[13] http ://www.securiteinfo.com/conseils/psiem.shtml

[14] D.BRENT Chapman et Elizabeth D .ZWICKY. "Building Internet Firewalls ", 2nd Edition, France, 1997.

[15] Bruce SCHNEIER. *"Cryptographie Appliquée, protocoles algorithmes et codes sources"*. Edition Vuibert, France, 1994.

[16] Jai Sundar Balasubramaniyan, Jose Omar Garcia-Fernandez, David Isacoff, Eugene Spafford, and Diego Zamboni. "An architecture for intrusion detection using autonomous agents". In Proceedings of the Fourteenth Annual Computer Security Applications Conference, pages 13-24. IEEE Computer Society, December 1998.

[17] Alban Jacquemin et Adrien Mercier. "*Les fire-walls*". sécurité de l'entreprise, France le 15 Février 2004.

[18] M. Pirio. *Appache(version 2) Installation, administration et sécurisation*. Editions ENI, Janvier 2004.

[19] Mark LUDWI. "Du Virus à l'antivirus". Edition Dunod Paris, 12 Mai 1997.

[20] C. Queinnec H. Schauer L. Bloch, C. Wolfhugel avec la contribution de N. Makarevitch. *Sécurité informatique*. Collection Blanche, mai 2009.

[21] P. Agrawal, R. K. Ghosh, and S. K. Das. Cooperative black and grey hole attack in mobile ad hoc networks. In Proceedings of the 2nd International Conference Ubiquitous Information Management and Communication (ICUIMC'08), pages 310 314, New York, NY, USA, 2008. ACM.

[22] A. ZEBDI : DZ-MAODV."nouveau protocole de routage multicast pour les réseaux ad hoc mobiles basé sur les zones denses". Avril 2006.

[23] B.jouga, J. marc : détection d'intrusion dans les réseaux ad hoc. 2003

[24] Yingshu Li, My T. Thai, Weili Wu, "Wireless Sensor Networks and Applications ", Springer Science+Business Media LLC, 2008

[25] Chris Karlof, David Wagner, "Secure routing in wireless sensor networks : attacks and countermeasures", Ad Hoc Networks 1(2003) 293 ?315, 2003

[26] http//www.rennes.supelec.fr/ren/perso/jlebegue/docs/crisis05. Le 19 février 2013.

[27] Ruiz, Friginal, David de-Andrés, Pedro Gil "Black Hole Attack Injection in Ad hoc Networks", Polytechnic University of Valencia, Campus de Vera s / n, E-46022, Valencia, Spain.

[28] R- Milner. "Communication and concurrency". Prentice Hall International (UK) Ltd, Hcrtfordshirc, UK, UK, 1995.

[29] Robin Milner. "Communicating and Mobile Systems : the Pi-Calculus". Cambridge University Press, June 1999.

[30] C. A. R. Hoare. "A model for communicating sequential processes" On the Construction of Programs, R. M. McKeag and A. M. McNaughton, Eds. London, England : Cambridge University Press, 1980, pp. 229-243.

[31] J.A. Bergstra and J.W. Klop. "Process algebra for synchronous communication". Technical Report 60, 1984.

[32] K.I.Consulting K.Ingham and S.Forrest. "A history and survey of network firewalls". The University of New Mexico Computer Science Department Technical Report, 2002.

[33] T. Mechri. "Approche algébrique pour la sécurisation des réseaux informatiques ". Mémoire, Université Laval, Québec. 2007.

[34] C. A. R. Hoare. "Communicating sequential processes". Communications of the ACM, 21 :666-677, 1985.

[35] J. Parrow et D. Walker, R. Milner." A calculus of mobile processes", i. In Inf.Comput, 100(1), pages 1-40.

[36] J. Parrow et D. Walker, R. Milner." A calculus of mobile processes", ii. In Inf.Comput, 100(1), pages 41-77.

[37] Robin Milner. "Communicating and Mobile Systems : the Pi-Calculus". Cambridge University Press, June 1999.

[38] J Parrow. "An introduction to the pi-calculus. In Dans Handbook of process algebra", Bergstra, Ponse et Smolka éditeurs, Elsevier, pages 479-543, 2001.

[39] D.Sangiori et D.Walker. "The pi-calculus : A theory of mobile processes". In Cambridge University Press, 2003.

[40] L. Cardelli and A. D. Gordon. "Mobile ambients". In Foundations of Software Science and Computation Structures : First International Conference, FOSSACS 98. Springer-Verlag, Berlin Germany, 1998.

[41] A. Cortesi and R. Focardi. "Information flow security in mobile ambients". Electronic Notes in Theoretical Computer Science, 54, 2001.

[42] F. Nielson, H. R. Nielson, R. R. Hansen and J. G. Jensen. "Validating firewalls in mobile ambients". In International Conference on Concurrency Theory, pages 463-477, 1999.

[43] Pierpaolo Degano, Francesca Levi and Chiara Bodei. "Safe ambients : Control flow analysis and security". In ASIAN 00 : Proceedings of the 6th Asian Computing Science Conference on Advances in Computing Science, pages 199-214, London, UK, 2000. Springer-Verlag.

[44] Francesca Levi and Davide Sangiorgi. "Controlling interference in ambients". In Symposium on Principles of Programming Languages, pages 352-364, 2000.

[45] Francesca Levi and Davide Sangiorgi. "Mobile safe ambients". ACM Trans. Program. Lang. Syst, 25(1) :1-69, 2003.

[46] Michele Bugliesi and Giuseppe Castagna. "Secure safe ambients". In POPL 01 : Proceedings of the 28th ACM SIGPLAN-SIGACT sympo-sium on Principles of programming languages, pages 222-235, New York, NY, USA, 2001. ACM.

[47] G. Ferrari, E. Moggi, and R. Pugliese. "Guardians for ambient-based monitoring". F-WAN : Foundations of Wide Area Network Computing, 66, 2002.

[48] D. Teller, P. Zimmer, and D. Hirschkoff. "Using ambients to control resources". International Journal of Information Security, 2 :126-144. Springer, 2004.

[49] K.ADI, L.HAMZA, L.PENE."Formal modeling for security behavior analysis of computer systems", IEEE Montreal conferenece on E technologies.23 25 janvier 2008. Canada.

[50] K.ADI, L.HAMZA, L.PENE."Approche algébrique pour le renforcement de la politique de sécurité dans les systèmes informatique" , doctorial , JDI'10, BEJAIA. Octobre 2010.

[51] Y.Zhang, W.Lee :"Intrusion detection in wireless ad hoc networks". *Proceedings of the 6th ACM International Conference on Mobile Computing and Networking,(MobiCom'00)2000.*

[52] Y.Zhang, W.Lee et Y.Huang :Inrusion detection techniques for mobile wirelessnetworks. *ACM/Kluwer Mobile Networks and Applications(MONET),to appear,2002.*

[53] Mohammad Al-Shurman, Seong-Moo Yoo et Seungjin Park ."Black Hole Attack in Mobile Ad Hoc Networks". In Proceedings of the 42nd Annual Southeast Regional Conference, 2004, Huntsville, Alabama, USA, April 2-3,2004.

[54] M.Saint-Laurent :" Genspec : Rapport final. environnement graphique pour la spécification de propriétés des systèmes informatiques". Technical report, Université du Québec en Outaouais, Gatineau, Québec (2009).

[55] http ://www.e-citz.com/eclipse.html. Le 18 mai 2013.

Annexe A

Annexe

Exemple 1 : Modélisation de l'attaque IP spoofing

1.1 Notions supplémentaires sur l'usurpation d'adresse IP

En générale, pour envoyer un message à une machine du même sous-réseau, il faut notamment son adresse MAC et pour avoir cette adresse, if faut vérifier d'abord son existence dans le cache ARP (Adress Resolution Protocol). Si elle n'est pas présente, on envoie un paquet ARP_REQUEST à toutes les machines et on demande qui a l'adresse MAC associée à cette adresse IP ?. La machine en question nous répond et le cache ARP sera mis à jour. Nous pouvons alors obtenir l'adresse MAC de la machine distante et lui envoyer le paquet. L'attaque étudiée consiste à ne pas attendre qu'une machine demande l'adresse MAC. Mais, l'intrus peut très bien la lui communiquer à n'importe quel moment en lui envoyant un simple paquet ARP_REPLY . Cela mettra à jour son cache ARP.

Nous prenons un exemple d'un réseau ou l'intrus corrompe le cache ARP du client en y inscrivant la correspondance entre son adresse MAC et l'adresse IP du routeur, tous les paquets qui transitent de la machine client au routeur seraient alors interceptés par l'attaquant. Cela permettrait notamment d'intercepter toutes les requêtes émises sur Internet par la machine cible.

1.2 Spécification et modélisation de l'attaque IP Spoofing

Notre exemple d'application, montré dans la figure A.1, qui est un système simple représenté par deux zones logiques : Internet et un réseau LAN vulnérable a une attaque de Spoofing IP . Pour rendre l'exemple plus facile à comprendre le nombre de machines est restreint. Le réseau LAN contient deux ordinateurs (un pour le client et l'autre pour l'intrus), un switch et un routeur pour que l'entreprise se connecte à

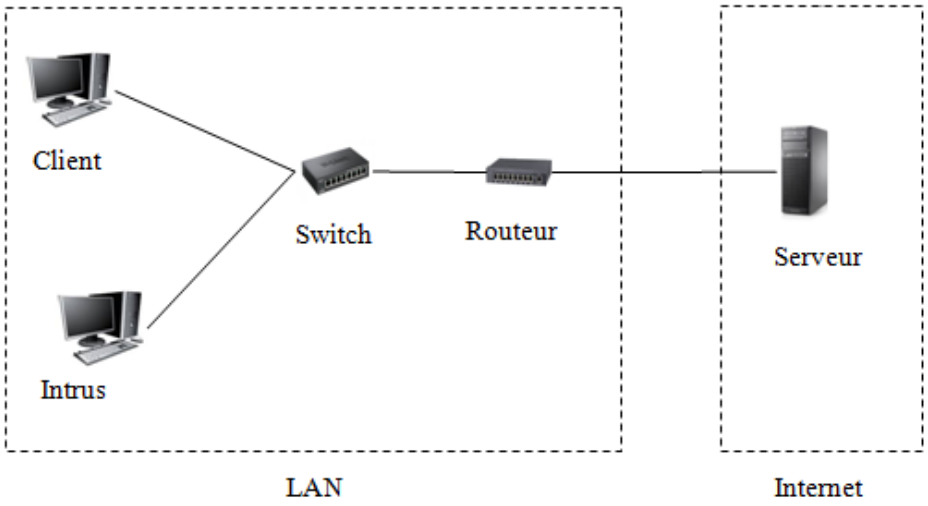

FIGURE A.1: Topologie d'un réseau vulnérable à une attaque d'IP spoofing

l'internet. La zone Internet contient un seul serveur de données . Chaque machine de l'exemple (à l'exception du switch) est représentée par un processus fonctionnant dans un ambient protégé, ainsi que le réseau LAN. Les noms d'ambients et de processus sont basés sur la première lettre du nom de l'ordinateur. Par exemple, l'ambient client est nommé c et contient le processus C, et ainsi de suite. Dans le calcul de [47, 48], chaque ambient est protégé par une clef. Une clef peut avoir la valeur publique par défaut δ ou une valeur différente secrète imposée par la politique de sécurité. La valeur δ assignée à la clef déclare qu'il n'y a pas de restriction pour l'action d'entrée ou de sortie. La hiérarchie des ambients et des sous-ambients reflète la topologie du réseau représentée sur la figure A.1. Ainsi les ambients l et s sont des ambients de niveau supérieur. Nous spécifions trois sous ambients dans l'ambient l : c, i et r, ainsi qu'un service de publication de clefs vers l'ambient c. L'intrus peut recevoir un ARP_REQUEST de la part du client et ensuite attaquer et usurper l'identité du routeur en envoyant un ARP_REPLY qui contient l'adresse MAC falsifiée et cela avant que le routeur envoie sa réponse.
Le système qui représente le réseau est indiqué par le processus suivant :

$$\mathrm{S} = {}^{\delta}_{l}[\mathrm{L} \mid !(pub^{k1}_{c}.0) \mid {}^{k1}_{c}[\mathrm{C}] \mid {}^{\delta}_{i}[\mathrm{I}] \mid {}^{\delta}_{r}[\mathrm{R} \mid !(pub^{k2}_{s}.0)]] \mid {}^{k2}_{s}[\mathrm{S}]$$

1.3 Processus d'intrus

Le processus d'intrus représenté par l'envoie du message ARP_REPLY dont le but est d'effectuer une attaque de spoofing d'adresse IP. L'intrus connait au commencement les clefs des ambients l et i. l'opérateur exp modélise la manière dont l'intrus explore le système et gagne des capacités en termes d'action mov, in ou out. Cet opérateur est le constructeur syntaxique qui imite le comportement non déterministe de l'intrus et il donne à l'intrus le choix d'employer ces clefs et de lancer son exploration.
Le processus d'intrus est modélisé comme suit :
I = $mov_i^\delta \oplus exp$.I'
Il y a quatre comportements possibles de l'intrus présenté par l'opérateur $\oplus$ dans l'expression :
$mov_i^\delta \oplus exp = mov_i^\delta \sqcap exp \sqcap mov_i^\delta.exp \sqcap exp.mov_i^\delta$
Le meilleur comportement que l'intrus peut choisir est : $mov_i^\delta.exp$.I' , les autres choix sont ignorés parcequ'ils produisent aucun résultat ou ils paralysent l'attaque.
Dans un premier temps le processus d'intrus sort de l'ambient i par l'exécution de l'action mov_i^δ ensuite il poursuit son exploration dans le système. Le tableau A.1 montre les étapes de l'exploration.

$\rightarrow$	${}_l^\delta$[L $\mid !(pub_c^{k1}.0) \mid {}_c^{k1}$[C] $\mid {}_i^\delta[mov_i^\delta \oplus exp$.I'] $\mid {}_r^\delta$[R $\mid !(pub_s^{k2}.0)]] \mid {}_s^{k2}$[S]
$\rightarrow$	${}_l^\delta$[L $\mid !(pub_c^{k1}.0) \mid {}_c^{k1}$[C] $\mid {}_i^\delta[mov_i^\delta.exp$.I'] $\mid {}_r^\delta$[R $\mid !(pub_s^{k2}.0)]] \mid {}_s^{k2}$[S]
$\rightarrow$	${}_l^\delta$[L $\mid exp$.I' $\mid !(pub_c^{k1}.0) \mid {}_c^{k1}$[C] $\mid {}_i^\delta$[0] $\mid {}_r^\delta$[R $\mid !(pub_s^{k2}.0)]] \mid {}_s^{k2}$[S]
$\rightarrow$	${}_l^\delta$[L $\mid mov_c^{k1} \oplus exp$.I' $\mid !(pub_c^{k1}.0) \mid {}_c^{k1}$[C] $\mid {}_i^\delta$[0] $\mid {}_r^\delta$[R $\mid !(pub_s^{k2}.0)]] \mid {}_s^{k2}$[S]
$\rightarrow$	${}_l^\delta$[L $\mid mov_c^{k1}.exp$.I' $\mid !(pub_c^{k1}.0) \mid {}_c^{k1}$[C] $\mid {}_i^\delta$[0] $\mid {}_r^\delta$[R $\mid !(pub_s^{k2}.0)]] \mid {}_s^{k2}$[S]
$\rightarrow$	${}_l^\delta$[L $\mid !(pub_c^{k1}.0) \mid {}_c^{k1}$[C $\mid exp$.I'] $\mid {}_i^\delta$[0] $\mid {}_r^\delta$[R $\mid !(pub_s^{k2}.0)]] \mid {}_s^{k2}$[S]

TABLE A.1: Comportement du processus d'intrus

1.4 Résultat d'exécution de l'exemple 1

La figure A.2 montre la topologie de l'exemple présenté précédemment, et la figure A.3 présente le résultat finale de l'exploration de l'intrus.

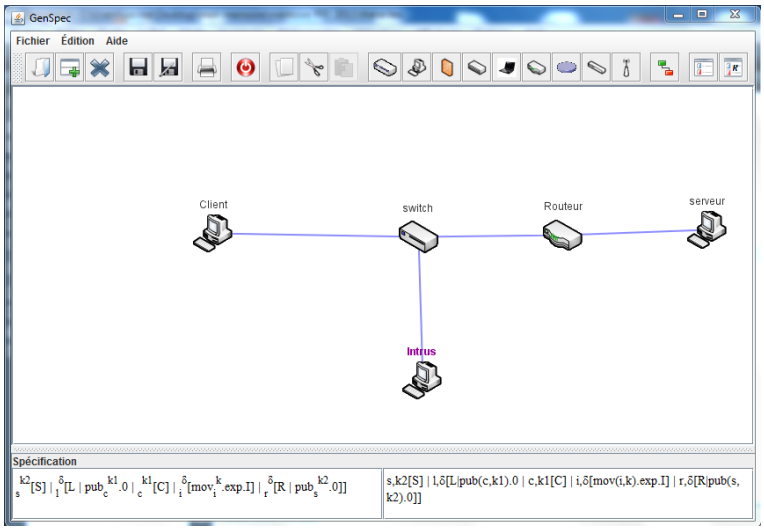

FIGURE A.2: Représentation du réseau sur l'application

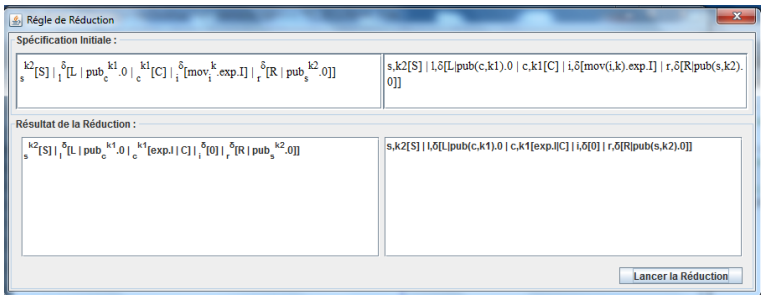

FIGURE A.3: Resultat final d'exploration de l'intrus

Exemple 2 : Modélisation d'un réseau simple

2.1 Spécification et modélisation d'un réseau simple

Le cas étudier dans [47] est un système simple composé de trois ordinateurs connectés via un routeur : un poste de travail (A) et deux serveurs (B et C), tous avec leurs propres politiques de sécurité mises en œuvre(Figure A.4). Le client SSH s'exécute dans A. Le serveur SSH s'exécute dans B, avec un client FTP. Le serveur FTP et une base de données s'exécutent dans C. Les politiques de sécurité mises en place assurent une protection avec une clef pour chaque machine. L'accès à A est protégé par la clef d'accès k1, est régie par le mécanisme d'authentification intégré dans le système d'exploitation exécuté sur le poste de travail. La clef d'accès k3 résume les conditions nécessaires pour entrer dans B : la connaissance de l'adresse du serveur SSH dans B et les informations d'identification appropriées pour se connecter à ce serveur (une combinaison

nom d'utilisateur / mot de passe valide). La clef d'accès à B est connue par tous les processus de A.

L'accès à C est donné aux processus qui ont la clé K5, celle-ci regroupe les connaissances nécessaires pour accéder au serveur FTP, il contient : l'adresse du serveur, le nom d'utilisateur et un mot de passe. Seuls quelques processus de B ont ces informations. L'accès à la base de données DB ne nécessite aucun privilège spécial et il est disponible pour tous les utilisateurs authentifiés par le serveur de C. Certains processus n'ont aucune importance dans cet exemple. Afin de simplifier la présentation, les auteurs de [47] représentent tous les processus qui ne sont pas intéressants par un processus abstrait dans chaque machine.

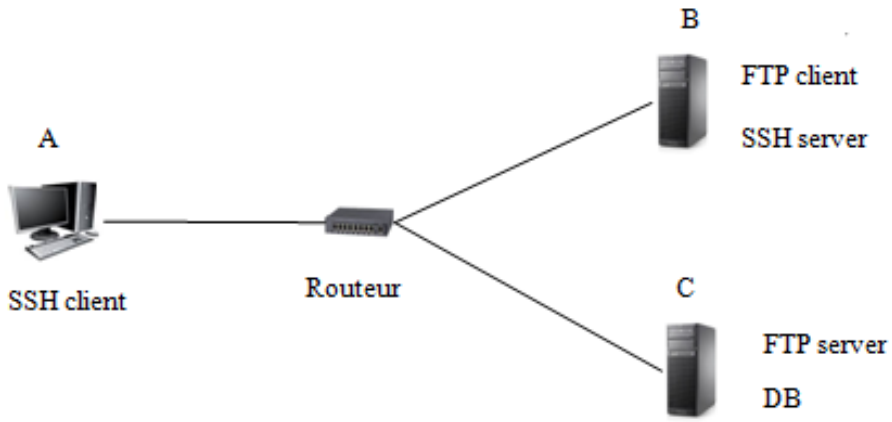

FIGURE A.4: Topologie d'un réseau simple

Chaque machine de l'exemple est représentée par un processus fonctionnant dans un ambient protégé, Les noms d'ambients et de processus sont basés sur la première lettre du nom de la machine. Par exemple, l'ambient de la machine A est nommé a et contient le processus A. Ceci est illustré par l'expression suivante :

$\overset{k1}{a}[\mathrm{A}] \mid \overset{k3}{b}[\mathrm{B}] \mid \overset{k5}{c}[\mathrm{C}]$

Le processus A est considéré comme la composition parallèle de deux processus M et A'. Le processus M dénote le processus régulier dans la machine A alors que A'

représente le reste de processus qui s'exécute dans cette machine.
Il existe un serveur fonctionnant dans la machine B, représenté comme un service de publication de clefs à l'intérieur de l'ambient protégé b.
Le processus de la machine C dans l'ambient c est la composition de deux processus T et C', où T représente une base de données d'informations sensibles qui ne devrait être disponible que pour certains processus de B. Ce processus est sélectionné pour évaluer si les processus réguliers et d'intrus peuvent accéder à la base de données ou s'ils sont conformes à la politique de sécurité qui refuse l'accès à ces données pour tous les utilisateurs de A. la même observation s'applique également à B et son serveur de publication de clefs. La clef K5 représente la vulnérabilité du système, celle-ci donne l'accès à tous les utilisateurs authentifiés dans le serveur SSH. Pour cette étude de cas, les auteurs montrent comment l'intrus peut entrer dans c et d'exécuter certaines actions en parallèle avec T. Cela correspond clairement à une violation de la politique de sécurité mise en œuvre. La spécification du système est décrite par le processus suivant :
$\overset{k1}{a}$[M|A'] | $\overset{k3}{b}$[B'| !(pub_c^{k5}.0)] | $\overset{k5}{c}$[T|C']

2.2 Le processus d'intrus

Nous appliquons la même technique présentée dans le chapitre 3 pour le processus de l'intrus. L'opérateur *exp* aidera l'intrus pour explorer le système et acquérir des capacités en terme d'actions *mov*,*in* ou *out*. Comme indiqué dans la section précédente tous les processus de A, y compris le processus de l'intrus connaissent la clef d'accès k3 vers l'ambient b. L'intrus a l'option d'utiliser les clefs qu'il connait au commencement puis lance son exploration. Il gagnera de nouvelles capacités de mouvement à partir des clefs publiées dans le système via des serveurs de publication (sans faire de demande). Le processus de l'intrus est illustré par l'expression suivante :

$$\text{I} = mov_c^{k1} \oplus mov_b^{k3} \oplus exp.\text{I'}$$

Il ya quinze comportements possibles de l'intrus présentés par les deux opérateurs

$\oplus$ dans l'expression ci-dessus, considérant le cas simplifié de a, b et c, les choix sont : $a \oplus b \oplus c = a \sqcap b \sqcap c \sqcap a.b \sqcap a.c \sqcap b.a \sqcap b.c \sqcap c.a \sqcap c.b \sqcap a.b.c \sqcap b.a.c \sqcap a.c.b \sqcap b.c.a \sqcap c.a.b \sqcap c.b.a.$ un seul comportement qui est utile dans son attaque, à savoir $mov_c^{k1}.mov_b^{k3}.exp.\text{I'}$. Les autres choix sont ignorés, parcequ'ils produisent aucun résultat ou ils paralysent l'attaque. Le tableau A.2 montre les étapes de l'exploration de l'intrus.

$\rightarrow$	$\overset{k1}{a}[mov_c^{k1}.mov_b^{k3}.exp.\text{I'}\vert\text{A'}] \mid \overset{k3}{b}[\text{B'}\vert !(pub_c^{k5}.0)] \mid \overset{k5}{c}[\text{T} \mid \text{C'}]$
$\rightarrow$	$\overset{k1}{a}[\text{A'}] \mid mov_b^{k3}.exp.\text{I'} \mid \overset{k3}{b}[\text{B'}\vert !(pub_c^{k5}.0)] \mid \overset{k5}{c}[\text{T} \mid \text{C'}]$
$\rightarrow$	$\overset{k1}{a}[\text{A'}] \mid \overset{k3}{b}[exp.\text{I'} \mid \text{B'}\vert !(pub_c^{k5}.0)] \mid \overset{k5}{c}[\text{T} \mid \text{C'}]$
$\rightarrow$	$\overset{k1}{a}[\text{A'}] \mid \overset{k3}{b}[mov_c^{k5} \oplus exp.\text{I'} \mid \text{B'}\vert !(pub_c^{k5}.0)] \mid \overset{k5}{c}[\text{T} \mid \text{C'}]$
$\rightarrow$	$\overset{k1}{a}[\text{A'}] \mid \overset{k3}{b}[mov_c^{k5}.exp.\text{I'} \mid \text{B'}\vert !(pub_c^{k5}.0)] \mid \overset{k5}{c}[\text{T} \mid \text{C'}]$
$\rightarrow$	$\overset{k1}{a}[\text{A'}] \mid \overset{k3}{b}[\text{B'}\vert !(pub_c^{k5}.0)] \mid \overset{k5}{c}[exp.\text{I'} \mid \text{T} \mid \text{C'}]$

TABLE A.2: Les étapes d'exploration de l'intrus

1.4 Résultat d'exécution de l'exemple 2

La figure A.5 montre la topologie du réseau, et la figure A.6 présente le résultat finale de l'exploration de l'intrus.

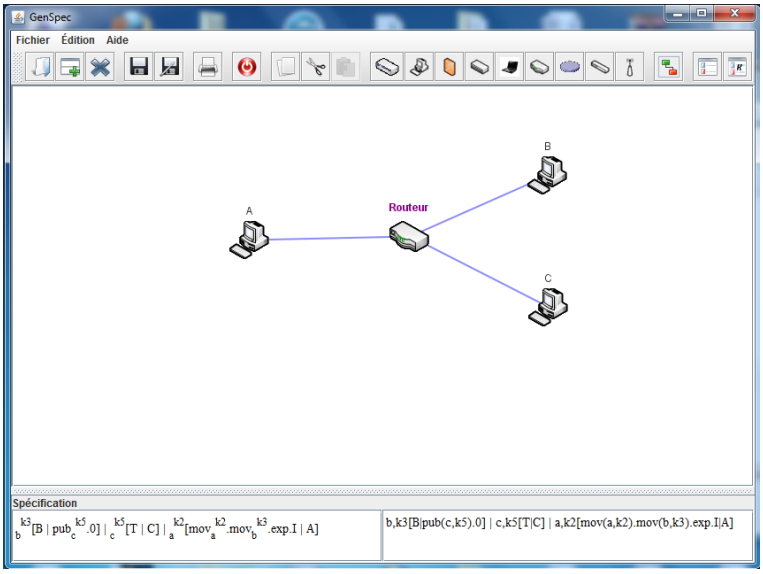

FIGURE A.5: Représentation du réseau sur l'application

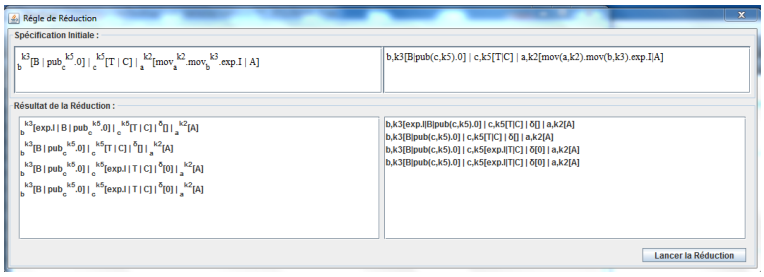

FIGURE A.6: Résultat final de l'exploration de l'intrus

Résumé

Aujourd'hui, aucun système informatique n'est sûr à cent pour cent. Configurer un système informatique afin de le rendre sécurisé est une tache complexe, elle se complique plus avec l'ampleur du système à protéger. De ce fait il est important de développer des méthodes formelles permettant de configurer d'une manière sécurisée les systèmes informatiques. L'objectif de ce projet est la modélisation d'une attaque dans les réseaux AD HOC sous la forme d'un langage formel avec l'application de l'approche RSC, cela nous permet d'obtenir une version plus sécurisé du réseau.

Mots clés *: Sécurité informatique, Réseaux AD HOC, Méthodes formelles, Algèbre de processus, Calcul ambient.*

Abstract

Nowdays, no computer system is secure at a hundred percent. Configure a computer system to make it secure is a complex task, it is more difficult with the size of the system to be protected. Therefore it is important to develop formal methods for configuring a secure way computer systems. The objective in this project is the modeling of an attack in the AD HOC networks as a formal language with the application of the RSC approach, this allows us to obtain a more secure version of the network.

Key-words *:Computer Security, AD HOC Networks, Formal Methods, Process algebra, Calculus ambient.*