THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par *l'Université Toulouse III - Paul Sabatier*
Discipline ou spécialité : *Informatique et Télécommunications*

Présentée et soutenue le *15/09/2014* **par :**
Hicham EL KHOURY

Titre : *Une modélisation formelle orientée flux de données pour l'analyse de configuration de sécurité réseau*

Jury

Président : *Mme. Francine KRIEF, Professeur, ENSEIRB, Bordeaux*
Rapporteurs : *M. Ahmed SERHROUCHNI, Professeur, TELECOM-Paris, Paris*
M. Abdelmadjid BOUABDALLAH, Professeur, Université de Technologie Compiègne
Examinateurs :
M. Abdelmalek BENZEKRI, Professeur à l'Université Toulouse III, Toulouse
M. Ibrahim MOUKARZEL, Professeur à l'Université Libanaise, Liban
M. François BARRERE, Maître de conférences à l'Université Toulouse III, Toulouse
M. Maroun CHAMOUN, Maître de conférences à l'Université Saint Joseph, Liban
M. Romain LABORDE, Maître de Conférences à l'Université Toulouse III, Toulouse

---

Ecole doctorale : *ED MITT : Mathématiques Informatique Télécommunications de Toulouse*
Unité de recherche : *IRIT-UMR 5505*
Directeur(s) de Thèse : *Pr. Abdelmalek BENZEKRI*
Co-Encadrants : *M. François Barrère*
M. Romain Laborde

A mes parents et mes frères Bassem et Wissam

A ma femme Sandra et mon fils Jad

« La communication est la clé de la réussite »

Remerciements

« *Quand vient le moment de la récolte, ne jamais oublier ceux qui vous ont supporté et prêté main forte lors de la semence* ».

Les travaux présentés dans ce mémoire ont été effectués au sein du Laboratoire de l'Institut de Recherche en Informatique de Toulouse (IRIT) dans l'équipe d'administration de réseaux et intégration de services (SIERA). Je tiens à remercier les personnes qui m'ont aidé à réaliser cette thèse et à rédiger ce document.

Je remercie Monsieur **Ahmed Serhrouchni**, Professeur à Telecom ParisTech -Institut Mines-Télécom et Monsieur **Abdelmadjid Bouabdallah**, Professeur à l'Université de Technologie de Compiègne, pour avoir accepté la charge de rapporteurs de cette thèse et leur participation à ce jury. Je remercie également, Monsieur **Ibrahim Moukarzel,** Professeur à l'Université Libanaise, pour l'aide qu'il m'a accordé et en m'honorant de sa présence en qualité d'examinateur dans ce jury de thèse.

Je souhaite exprimer toute ma gratitude envers Madame **Francine Krief**, Professeur à l'ENSEIRB, membre du Laboratoire Bordelais de Recherche en Informatique et directrice générale adjointe de l'Institut Polytechnique de Bordeaux pour l'intérêt qu'elle a porté à ce travail.

Je remercie le Professeur **Abdelmalek Benzekri** pour la confiance qu'il m'a témoignée en m'accueillant dans son équipe de recherche, pour sa patience et le soutien constant dans la direction de cette thèse.

Je remercie **Romain Laborde** pour toute l'attention et la disponibilité dont il a fait preuve dans le co-encadrement de cette thèse, il m'a beaucoup appris ; sa compétence, sa clairvoyance, son charisme et son dynamisme ont été essentiels pendant toute la durée de cette préparation de thèse et pour son investissement dans la production de ce document.

Je remercie **François Barrère** pour tout le temps qu'il m'a consacré dans ce co-encadrement, pour son esprit cartésien, pour ses précieux conseils et son orientation ficelée tout au long de ce travail de recherche, et la part très active dans la production du document de thèse.

J'adresse également mes remerciements à **Maroun Chamoun** maître de Conférences à l'Université Saint Joseph - Liban, pour les conseils qu'il a pu me donner, l'expérience dont il a su me faire part dans ce domaine et enfin pour sa participation à ce jury en qualité d'examinateur.

Même si elles n'ont pas apporté une contribution sur le plan scientifique à cette thèse, je tiens à remercier toutes les personnes qui de près ou de loin m'ont aidé directement ou indirectement dans l'aboutissement de ce travail. En particulier, je remercie M. **Georges Rahbani,** Directeur de l'Université Libanaise faculté de science II et ainsi que M. **Naji Akiki**, le chef de département informatique, pour avoir compris l'importance de mes séjours à Toulouse, leur soutien, et pour avoir facilité l'organisation de ces déplacements. Je remercie également l'ensemble de mes collègues pour avoir accepté des modifications d'emploi du temps et avoir pris le relais pendant mes périodes de déplacements. Je remercie également les membres de l'école doctorale EDMITT, son directeur Monsieur RoqueJoffre ainsi que **Martine Labruyère** pour sa patience et ses rappels avisés.

Je veux aussi remercier tout l'équipe SIERA, les permanents et les doctorants. Spécialement mes camarades dans l'équipe, **Marwan Chaeito, Samer Wazan, Tony Toueir, Bachar Kabbani, Messaoud Aouadj, Rémi Venant, Pierrick Marie et Arnaud Oglaza** qui ont contribué à rendre mon séjour et mes études agréables et enrichissants.

Je souhaite exprimer ma gratitude à tous mes amis qui m'ont si bien encouragé, soutenu et aidé à accomplir jusqu'ici toutes les étapes de ma thèse, que j'espère finir avec succès. Vous avez étés mes « murs porteurs » : le prince 'zahlaouite' **Antoine Abou-Fayad,** le professionnel **Johny Matar et** les cousins **Mario et Randa Nader**.

Les travaux effectués durant cette année n'auraient pas été possible sans le soutien absolu de **mes frères : Bassem** et **Wissam** et surtout de **mes parents** qui par leurs prières et leurs encouragements, on a pu surmonter tous les obstacles. Je vous aime de tout mon cœur et je demeurerai toujours à vos côtés.

Avant de terminer, je réserve une reconnaissance particulière à ma femme **Sandra Koueik**, pour l'amour et le soutien incomparable dont elle a fait preuve depuis le 2 février 2009. A mon petit **Jad**, avec la plus grande réjouissance eu à sa naissance. Pour le bébé que nous attendons.

Pour clore ces remerciements, je tiens à exprimer ma sincère gratitude à tous ceux qui ont contribué à la concrétisation de cette thèse. Merci à tous d'avoir apporté une pierre à l'édifice et de m'avoir donné l'opportunité de faire un pas en avant vers l'avenir.

Résumé

La mise en œuvre d'une politique de sécurité réseau consiste en la configuration de mécanismes de sécurité hétérogènes (passerelles IPsec, listes de contrôle d'accès sur les routeurs, pare-feux à états, proxys, etc.) disponibles dans un environnement réseau donné. La complexité de cette tâche réside dans le nombre, la nature, et l'interdépendance des mécanismes à considérer. Si différents travaux de recherche ont tenté de fournir des outils d'analyse, la réalisation de cette tâche repose aujourd'hui encore sur l'expérience et la connaissance des administrateurs sécurité qui doivent maîtriser tous ces paramètres.

Dans cette thèse nous proposons une solution pour faciliter le travail des administrateurs réseau. En effet, nombre d'inconsistances viennent de l'incompatibilité de règles de politiques, de l'incompatibilité de mécanismes mis en œuvre successivement au sein des équipements traversés. Une théorie formelle générique qui permet de raisonner sur les flux de données réseau est manquante. Dans cette optique, nous présentons trois résultats complémentaires :

- un modèle formel orienté flux de données pour l'analyse de politiques de sécurité réseau afin de détecter les problèmes de consistance entre différents mécanismes de sécurité sur des équipements différents jouant un rôle à différents niveaux dans les couches ISO. Dans ce modèle, nous avons modélisé un flux d'information par un triplet contenant la liste des protocoles de communication dont le flux résulte, la liste des attributs dont l'authentification est garantie, et la liste des attributs dont la confidentialité est garantie.
- un formalisme indépendant de la technologie basé sur les flux de données pour la représentation des mécanismes de sécurité ; nous avons spécifié formellement la capacité et la configuration des mécanismes de sécurité devant être mis en œuvre en construisant une abstraction des flux physiques de blocs de données. Nous avons proposé une solution qui peut répondre aux exigences de sécurité et qui peut aider à analyser les conflits liés au déploiement de plusieurs technologies installées sur plusieurs équipements
- afin d'évaluer à la fois la capacité d'expression et d'analyse du langage de modélisation, nous avons utilisé les réseaux de Petri colorés pour spécifier formellement notre langage.

L'objectif de nos recherches vise l'intérêt et la mise à disposition d'un langage de modélisation pour décrire et valider les architectures solutions répondant à des exigences de sécurité réseau. Des simulations appliquées à des cas particuliers, comme le protocole IPsec, NA(P)T et Netfilter/iptables, complètent notre démarche. Néanmoins, l'analyse des conflits de sécurité se fait actuellement par simulation et de manière non exhaustive. Nos travaux futurs viseront à aider/automatiser l'analyse en permettant aux intéressés de définir les propriétés en logique temporelle par exemple qui seront contrôlées automatiquement.

Mots-clés : Sécurité des Réseaux ; Analyse des configurations ; Détection des conflits ; Modélisation et simulation ; Spécification formelle ; CPN (réseaux de Petri colorés) ;

Abstract

The implementation of network security policy requires the configuration of heterogeneous and complex security mechanisms in a given network environment (IPsec gateways, ACLs on routers, stateful firewalls, proxies, etc.). The complexity of this task resides in the number, the nature, and the interdependence of these mechanisms. Although several researchers have proposed different analysis tools, achieving this task still requires experienced and proficient security administrators who can handle all these parameters.

In this thesis, we propose a solution to facilitate the work of network administrators. Indeed, many inconsistencies come from the incompatibility of policy rules and/or incompatible mechanisms implemented in devices through which packets travel. A generic formal theory that allows reasoning about network data flows and security mechanisms is missing. With this end in mind, we develop in this thesis three results:

- A formal data-flow oriented model to analyze and detect network security conflicts between different mechanisms playing a role at various ISO levels. We modeled a flow of information by a triplet containing the list of communication protocols (i.e., encapsulation), the list of authenticated attributes and the list of encrypted attributes,
- A generic attribute-based model for network security mechanisms representation and configuration. We have formally specified the capacity and configuration of security mechanisms by constructing an abstraction of physical flows of data blocks. We have proposed a solution that can satisfy security requirements and can help conflicts analysis in the deployment of technologies installed on different devices,
- To evaluate both the ability of expression and analysis power of the modeling language. We have used CPN Tools [Jensen et Kristensen 2009] and [CPN tools] to formally specify our language.

The goal of our research is to propose a modeling language for describing and validating architectural solutions that meet network security requirements. Simulations are applied to specific scenarios, such as the IPsec, NA(P)T and Netfilter/iptables protocols, to validate our approach. Nevertheless, the analysis of security conflicts is currently done by simulation and in a non-exhaustive manner. Our future work will aim to assist/automate the analysis by allowing the definition of properties in temporal logic for instance which will be automatically controlled.

Keywords : Network Security; Configurations analysis; Conflict Detection; Modeling; Formal Specification; CPN (Colored Petri Nets).

Liste de Publications

- ***Articles de revues internationales***

  - Hicham El Khoury, Romain Laborde, François Barrère, Maroun Chamoun, Abdelmalek Benzekri. "A Data Flow Oriented Specification Method for Analyzing Network Security Configurations". *Dans : International Journal of Internet Protocol Technology, Accepté.*

- ***Conférences et workshops internationaux :***

  - François Barrère, Romain Laborde, Hicham El Khoury, Abdelmalek Benzekri,. "A formal data flow model to study network protocol encapsulation inconsistencies" *(poster). Dans : WORLDCOMP'14 (SAM 2014), Las Vegas Nevada, USA, July 21-24, 2014. CSREA Press ISBN: 1-60132-285-2, p. 411-412, 2014.*

  - Hicham El Khoury, Romain Laborde, François Barrère, Abdelmalek Benzekri, Maroun Chamoun. "A Specification Method for Analyzing Fine Grained Network Security Mechanism Configurations" *(short paper). Dans : Symposium on Configuration Analytics and Automation (SafeConfig 2013), Washington, D.C., USA, 16/10/13, (Eds.), IEEE, p. 483-487, 2013.*

  - Hicham El Khoury, Romain Laborde, Maroun Chamoun, François Barrère, Abdelmalek Benzekri. "A Generic Attribute-Based Model for Network Security Mechanisms Representation and Configuration" *(regular paper). Dans : International Conference on Frontier of Computer Science and Technology (FCST 2012), Suzhou, China, 21/11/12-23/11/12, Soochow University, (support électronique), novembre 2012.*

  - Hicham El Khoury, Romain Laborde, François Barrère, Maroun Chamoun, Abdelmalek Benzekri. "A Formal Data Flow-Oriented Model For Distributed Network Security Conflicts Detection" *(regular paper). Dans : International Conference of Networking and Services (ICNS 2012), St. Maarten, The Netherlands Antilles, 25/03/12-30/03/12, Xpert Publishing Service (XPS), p. 20-27, 2012.*

  - Hicham El Khoury, Romain Laborde, François Barrère, Abdelmalek Benzekri, Maroun Chamoun. "A Generic Data Flow Security Model" *(poster). Dans : Symposium on Configuration Analytics and Automation (SafeConfig 2011), Arlington, VA, USA, 31/10/11-01/11/11, IEEE, p. 1-2, 2011.*

- ***Conférences et workshops nationaux :***

  - Hicham El Khoury, Romain Laborde, François Barrère, Abdelmalek Benzekri. « Vers un modèle formel orienté flux de données pour l'analyse de politiques de sécurité réseau » *(regular paper). Dans : Conférence sur la Sécurité des Architectures Réseaux et Systèmes d'Information (SAR-SSI 2011), La Rochelle, 18/05/11-21/05/11, IEEE Computer Society - Conference Publishing Services, p. 1-7, 2011.*

Table des matières

Table des figures

Liste des tableaux

Chapitre 1. Introduction et Contributions

1.1 Motivations et enjeux

La sécurité des applications distribuées est assurée par un ensemble de services de sécurité. L'ISO définit les services suivants [ISO 7498:2] : le contrôle d'accès, l'identification/authentification, la confidentialité, l'intégrité, la non répudiation et la traçabilité. L'implémentation de ces services de sécurité au moyen de mécanismes de sécurité comme (IPSec, L2TP, SSL, SSH, PPPoE, etc.) ou encore de mécanismes de contrôle d'accès (les firewalls, les Application Level Gateways, les contrôles d'accès sur les systèmes) est l'une des priorités des architectures et infrastructures de communication. Des standards comme X.800 nous offrent un résumé de solutions de sécurité et montrent la diversité des technologies existantes et toujours actuelles.

Aujourd'hui, la tendance est à la virtualisation de l'infrastructure, ce qui permet de réduire les coûts informatiques tout en augmentant l'efficacité, le taux d'utilisation et la flexibilité des actifs existants. C'est pourquoi alors que l'on observe une migration d'un nombre important de services d'infrastructures physiques vers des infrastructures virtuelles, cette virtualisation rend l'implémentation des services de sécurité de plus en plus compliquée. En effet, si l'on considère, par exemple, une entreprise et ses filiales qui déploient des équipements interconnectés via un opérateur, ce dernier devra respecter le cahier des charges fourni et répondre à des exigences de cohérence et de compatibilité entre les équipements. Une fois cette étape franchie, l'opérateur appliquera sa propre politique d'interconnexion sans pour autant être intrusif ni permissif vis-à-vis de ses clients. C'est une autre facette de la complexité à différents étages et différents niveaux d'interconnexion. L'administrateur doit alors choisir sa solution de sécurité en composant selon ses besoins les services de sécurité réseau à utiliser et les configurations des mécanismes à appliquer dans un environnement où l'évolution continue et rapide des technologies augmente la complexité de cette tâche.

L'explosion des interconnexions des SI a entraîné le développement de solutions logicielles/matérielles de façon à répondre rapidement aux besoins et aux contraintes. Ces solutions se veulent dans beaucoup de cas opérationnelles mais ne satisfont pas les besoins de sécurité. Ceci est dû au fait que la plupart de ces solutions sont limitées, soit par le niveau d'abstraction élevé, soit par la limitation d'une ou de plusieurs technologies sous-jacentes. Il existe deux approches connues (cf. Figure 1) :

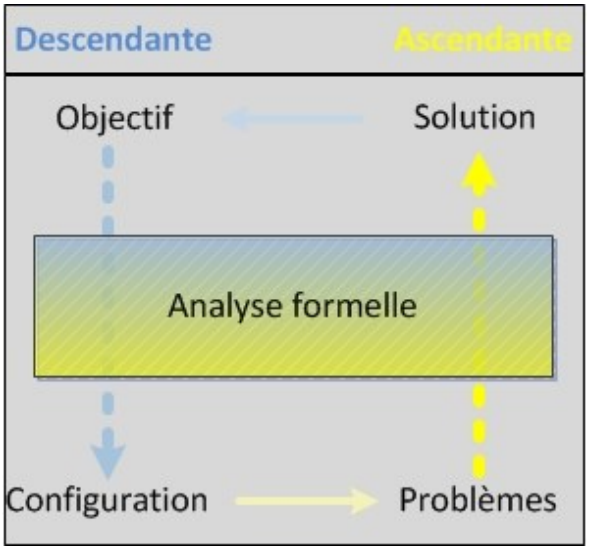

Figure 1. Niveau d'abstraction adéquat

- Approche Descendante (*Top/Down)* : Cette approche, qui est suivie par les praticiens de la gestion de réseau, consiste à utiliser différents niveaux d'abstraction de l'information de gestion qui aide les administrateurs à traduire leurs objectifs par une configuration ; c'est le processus de raffinement. Par exemple, le PBNM (Policy Based Network Management) utilise cette approche pour automatiser la tâche de gestion. Cependant, le processus de raffinement, est incomplet et n'est pas maitrisé encore. De plus, l'automatisation de ce type de solution est encore théorique [Laborde 2005].

- Approche Ascendante (*Bottom/up)* : Cette approche est utilisée pour diagnostiquer le mauvais fonctionnement des équipements réseaux. Elle consiste à analyser les configurations existantes sur les équipements de sécurité et en déduire l'exactitude (correctness) et la cohérence de ces configurations sur les équipements de réseau. Cette activité nécessite l'utilisation d'outils appropriés pour permettre aux analystes de sécurité de comprendre les interactions globales de toutes les configurations ; à titre exemple, les modèles sont liés à une ou plusieurs technologies précises et il est très difficile de les adapter à de nouvelles technologies [Al-Shaer et al. 2006].

Nous constatons que les deux approches atteignent partiellement leurs objectifs : le processus de raffinement de l'approche descendante est très théorique et ne s'implémente pas dans tous les cas, tandis que l'approche ascendante dépend du mécanisme à mettre en œuvre et du type d'équipement. Pour pouvoir analyser les politiques de sécurité indépendamment des équipements, nous avons formalisé le problème (*Analyse formelle* Figure 1). D'une part, nous avons conçu un modèle formel pour présenter les flux de données ce qui permet d'analyser les transformations d'un flux indépendamment des nœuds traversés et des mécanismes appliqués. D'autre part, nous avons défini formellement chaque mécanisme pour vérifier les configurations indépendamment des technologies sous-jacentes.

Les mécanismes qui permettent de mettre en œuvre et d'offrir les services de sécurité énumérés précédemment sont classés en deux catégories :

1) les mécanismes de chiffrement qui interviennent dans presque tous les mécanismes de signature numérique, des techniques d'utilisation d'identificateurs et de mots de passe.

2) les mécanismes de contrôle d'accès sur les systèmes, les passerelles applicatives (Application Level Gateways) ou les firewalls.

Beaucoup de ces mécanismes peuvent apparaître similaires par leurs noms ou par leurs formes et à l'opposé leurs composantes peuvent être différentes (e.g., un Firewall Cisco, une machine Linux avec iptables ou ipchains sont tous des Firewalls mais n'ont pas les mêmes mécanismes). Ainsi, ces mécanismes ne sont pas les mêmes, ce qui nous conduit à les définir formellement pour les modéliser et étudier les propriétés de sécurité pour répondre aux questions suivantes : Qu'y-a-t-il de commun entre ces mécanismes ? Et comment peut-on les représenter/observer de manière générique ?

Finalement, le fait de garantir les propriétés d'intégrité et de confidentialité, lors de la réalisation d'échanges de données, pose un problème sur le plan de l'investigation. Ce dernier est d'autant plus important que l'organisation des échanges met le plus souvent en œuvre des infrastructures qui sont gérées par des tiers multiples que ce soit dans des échanges nationaux ou internationaux. La complexité qui en découle est à différents niveaux : peut-on garantir les propriétés dans un tel contexte ? Quels sont les points de blocage que l'on peut rencontrer, que ce soit dans un réseau géré par une seule entité administrative ou par plusieurs ? Peut-on modéliser l'enchaînement des communications et des mécanismes choisis pour valider à l'avance le succès des échanges ou pour localiser des points de blocage ?

Ces questions auxquelles nous allons apporter des réponses ont pour objectif de perfectionner l'implémentation que l'on peut exploiter dans le cadre de la définition, la vérification et la validation de politiques de sécurité.

Bien sûr, il n'est pas possible de spécifier une méthode générique pour représenter un flux ou un mécanisme avec le langage humain trop ambigu. En fait, il existe des langages spécialisés : *les langages de spécification*. Un langage de spécification est un modèle de description de politiques indépendant du domaine d'application.

1.2 Contribution

La modélisation et le raffinement proposés par les travaux existants dans les deux approches étant limités, nous posons alors les questions suivantes :

- Comment trouver le niveau d'abstraction adéquat (Figure 1) pour à la fois être indépendant des mécanismes de sécurité et représenter de manière la plus fidèle la réalité ?

- Comment modéliser un flux d'une manière générique ?
- Comment manipuler ce flux de données d'une manière unifiée ?
- Comment modéliser un mécanisme avec sa configuration pouvant manipuler ces flux de données ?

Pour répondre à ces questions, il est nécessaire d'utiliser un langage de modélisation qui soit unifié. Dans notre approche, nous proposons un langage de modélisation orientée flux de données de façon à ce que ce flux soit un point commun pour tous les mécanismes quel que soit le niveau d'implémentation (à n'importe quel niveau logique des couches ISO). Ensuite, nous généralisons ce langage afin de représenter les différents mécanismes réseau qui sont décrits comme des fonctions manipulant ces flux. La détection de conflits entre règles de sécurité réseau est effectuée en définissant des contraintes sur les flux de données et sur les mécanismes de sécurité. Notre travail a été exprimé dans le langage CPN (Colored Petri Nets), ce qui nous permet d'utiliser les outils associés pour automatiser une partie du processus d'analyse.

Les principales contributions de cette thèse sont les suivantes :

1. **Vers un modèle formel orienté flux de données pour l'analyse de politiques de sécurité réseau et pour détecter les inconsistances distribuées (multi-mécanismes, multi-couches ISO).**

La première contribution porte sur la définition d'une approche de modélisation mathématique orientée flux de données dont l'objectif est de détecter les problèmes d'inconsistance entre mécanismes de sécurité : un flux de données possède des caractéristiques propres, transite dans un réseau en traversant des nœuds. Chaque nœud examine un flux entrant et opère éventuellement des modifications avant de le relayer ou de le présenter. Afin de détecter les erreurs, une modélisation des flux et des nœuds a été réalisée : tout mécanisme de sécurité est modélisé à partir du niveau fonctionnel qu'il apporte. Cette abstraction permet de se détacher des caractéristiques des constructeurs, de prendre en compte leur diversité et leurs possibles évolutions. La modélisation des flux, de leurs transformations lors du passage à travers les nœuds nous apporte une connaissance fine et donc réaliste des inconsistances qui peuvent les affecter.

Plus précisément, un flux de données peut être vu comme un ensemble contigu ou non d'octets de taille variable véhiculé sur un réseau. Cette suite d'octets est décomposée en blocs logiques respectant une encapsulation de protocoles notée par « $\mathcal{E}$ ». Par exemple, un flux de données correspondant à une requête HTTP peut être vu comme <*bloc protocole Ethernet, bloc protocole IP, bloc protocole TCP, bloc protocole HTTP*>. Chaque protocole divise le bloc d'octets associé en champs, que nous appelons attributs, selon leur description. Par exemple, les informations de contrôle du protocole Ethernet sont réparties sur 14 octets (adresse MAC destination, adresse MAC source, identifiant du protocole encapsulé) au début de la trame et 4 octets pour le champ de contrôle à la fin.

Dans un objectif d'analyse, nous avons considéré l'ensemble des algorithmes de sécurité traitant les chaînes d'encapsulation de protocoles (par exemple, DES, 3DES, HMAC-SHA1, etc.) et le niveau de chiffrement. Pour cela, nous avons associé à l'encapsulation de protocoles deux ensembles AUTHN et CONF qui représentent les attributs du flux de données ayant été authentifiés et chiffrés respectivement. Par exemple, si le traitement consiste en l'utilisation de ESP (encapsulation security payload), cela revient à ajouter de nouveaux protocoles dans la chaîne d'encapsulation ainsi que de nouvelles instances dans la relation AUTHN et de nouveaux attributs dans le multi-ensemble CONF.

A partir de ces définitions des éléments de base, nous présentons l'ensemble des flux de données par : $\mathcal{F} \subseteq \mathcal{E} \times \text{AUTHN} \times \text{CONF}$ [El-Khoury et al. 2011 a].

Ensuite, nous validons notre démarche de modélisation sur des cas connus des administrateurs (par exemple, IPsec et le NAPT) afin de vérifier à la fois la capacité d'expression et d'analyse [El-Khoury et al. 2011 b].

2. Un formalisme indépendant de la technologie orienté flux pour la configuration et la représentation des mécanismes de sécurité

Dans notre deuxième contribution, nous proposons d'étendre le travail de modélisation des flux et des équipements pour aller vers une généralisation des résultats.

En particulier nous avons cherché à inventorier les opérations réalisées au sein des nœuds traversés par un flux transmis dans un réseau. La classification propose des opérations de consultation et d'extraction d'information, mais également des opérations de modification (transformation) du contenu d'un flux. Nous avons défini 9 opérations élémentaires (2 opérations d'extraction d'informations qui s'appliquent sur les protocoles et les attributs, et 7 opérations qui permettent la modification du contenu).

Tout nœud atteint ou traversé par un flux exécute un traitement (simple passage, blocage, transformation, etc.). Les actions appliquées dépendent de la nature du flux et des valeurs d'attributs le caractérisant ainsi que des règles qui régissent le fonctionnement du nœud.

Plus spécifiquement, ce traitement sur un flux de données est alors vu comme une fonction particulière de $\mathcal{F}$ dans $\mathcal{F}$. Chaque traitement selon la technologie rencontrée considère un ensemble d'attributs du flux de données en entrée (par exemple les attributs adresses IP source et destination et protocole transport du protocole IP et les attributs numéros de port source et destination du protocole TCP/UDP pour une passerelle IPsec), et peut être de bloquer, propager, transformer ou filtrer indépendamment des technologies utilisées. Brièvement, un équipement est spécifié par au moins une fonction élémentaire ou une combinaison de fonctions élémentaires représentant ses capacités de traitement sur les flux de données.

Le traitement sera effectué par un mécanisme particulier avec une configuration spécifique. Ainsi, ce mécanisme a une capacité aussi spécifique qui représente ce qu'il peut faire. Par exemple, un pare-feu peut filtrer les paquets en analysant les adresses IP, les champs ports, etc. Un proxy HTTP

peut modifier les messages HTTP en analysant un ensemble de champs HTTP. La deuxième composante d'un mécanisme spécifique est sa configuration. La configuration définit un comportement spécifique basé sur les caractéristiques de la capacité du mécanisme. En conséquence, nous définissons un mécanisme « M » par $CAPABILITY_M$ pour représenter la capacité spécifique et par $CONFIGURATION_M$ pour représenter une configuration spécifique pour un mécanisme spécifique [El-Khoury et al. 2012 a] :

$$\mathbf{M} = < \mathbf{CAPABILITY_M}, \mathbf{CONFIGURATION_M} >$$

$\mathbf{CAPABILITY_M} = < \boldsymbol{\Sigma_M}, \boldsymbol{A_M}, \boldsymbol{EXPR_M^A} >$ où les caractéristiques des capacités potentielles de M basées sur :

1) $\boldsymbol{\Sigma_M}$ est l'ensemble fini non vide de types rencontrés (e.g., @ips est un attribut qui représente l'adresse IP source d'un paquet IP)
2) $\boldsymbol{A_M} = \boldsymbol{A_M^F} \cup \boldsymbol{A_M^{ctx}}$ | A_M={a_i | 1 ≤ i ≤ k ou **k** ∈ ℕ et Type(a_i) ∈ Σ_M }; A_M^F est l'ensemble des attributs d'un flux de données F qui peuvent être récupérés par le mécanisme M, et A_M^{ctx} est l'ensemble des attributs de contexte retrouvés dans une règle de configuration du même mécanisme. Nous appelons « attributs de contexte », les attributs utilisés dans une règle de configuration et qui ne se trouvent pas dans le flux de données (e.g., « –o eth0 » ou les informations « stateful » dans une règle iptables)
3) $\boldsymbol{EXPR_M^A}$ est l'ensemble fini des expressions qui permettent d'évaluer A_M ; cet ensemble est fourni par le mécanisme de sécurité M

Nous définissons les éléments de configuration d'un mécanisme M par la liste des règles « $RULE_M^A$ » et un algorithme de résolution des conflits « CRA » où les deux sont basés sur un ou plusieurs éléments de la capacité d'un mécanisme spécial « $CAPABILITY_M$ » :

$$\mathbf{CONFIGURATION_M} \in \mathbf{RULES_M} \times \mathbf{CRA_M}$$

Une règle d'un mécanisme M consiste en un ensemble de contraintes sur des attributs A_M, conjointement avec un ensemble d'actions $ACTION_M$, de l'ensemble de toutes les actions possibles. Chaque règle r ∈ $RULE_M^A$ contient un ensemble de conditions et un ensemble d'actions :

$$\mathbf{RULE_M^A} \subseteq \mathbf{CONDITION_M^A} \times \boldsymbol{ACTION_M^A}$$

Les actions sont les traitements disponibles par les mécanismes implantés sur les équipements d'un réseau. Ces actions sont exprimées au moyen de commandes primitives de manipulation de flux de données que nous avons définies [El-Khoury et al. 2012 b] (2 commandes pour l'extraction d'informations qui s'appliquent sur les protocoles et les attributs, et 7 commandes qui permettent la modification du contenu d'un flux).

3. Une méthode de spécification pour l'analyse des configurations de mécanismes de sécurité réseau

Notre dernière contribution réside dans la conception d'un nouveau formalisme pour modéliser les mécanismes mis en œuvre au sein des nœuds. Cette formalisation prend en compte la capacité d'un mécanisme, les informations de configuration spécifiques à un nœud et des informations de configuration dépendant du flux. Cette opération de modélisation et l'implémentation associée permettent d'approcher plus rigoureusement la détection des anomalies. Notre objectif est de pouvoir analyser les flux de données et leurs propriétés tout en connaissant à tout moment la valeur de tout attribut.

Nous avons exprimé notre modèle dans le langage des CPNs hiérarchiques afin de valider notre approche et d'utiliser les outils d'analyse associés implantés dans le logiciel « CPN Tools » [CPN Tools]. Les réseaux de Petri Colorés (CPNs) [Jensen et Kristensen 2009] sont un langage de spécification formel composé d'un ensemble de jetons dont le type est représenté par une couleur, d'un ensemble de transitions, d'un ensemble de places possédant un domaine (qui définit les types de jetons qui peuvent être stockés dans la place) et d'un ensemble d'arcs reliant les places et les transitions. Il permet de créer des modèles formels de systèmes.

Alors, nous avons utilisé le logiciel « CPN Tools » dans le cadre de ces travaux pour modéliser et analyser nos propositions. Cet environnement de développement de CPN utilise une extension du langage ML [Harper 2011] pour spécifier par exemple les couleurs de jeton, les gardes, les fonctions sur les arcs. Il nous faut démontrer que nous pouvons représenter tout mécanisme quel que soit le niveau logique considéré. Nous avons défini un modèle CPN appelé GAM (pour Generic Attribute-based Model) comme bloc logique de base d'un mécanisme de sécurité réseau. Une technologie de sécurité est alors représentée par un GAM ou une combinaison de GAMs.

Avec une architecture flexible et modulaire, GAM a pu suivre l'impact sur les attributs de flux à chaque étape réalisée par les fonctions de transformation associées à un mécanisme ; plus encore il nous offre une possibilité d'analyse par simulation. En outre, des technologies plus complexes peuvent être présentées dans GAM (i.e., iptables partage les règles dans des chaînes qu'on perçoit comme des mécanismes qui manipulent les attributs d'un flux) [El-Khoury et al. 2013].

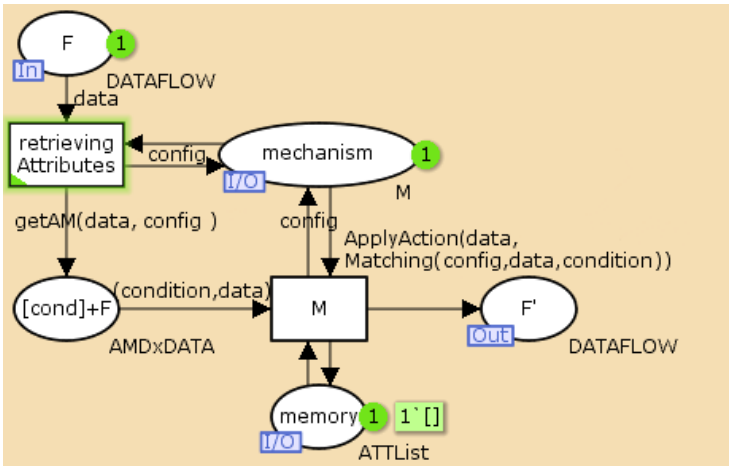

Figure 2. Présentation de GAM en CPN

Un GAM prend en entrée un DATAFLOW (place F dans la Figure 2 marquée par « In » en bleu) et retourne un DATAFLOW comme sortie (place F' dans la Figure 2 marquéee par « Out » en bleu). La description de ce mécanisme générique est définie dans la place « mechanism » qui contient la capacité et la configuration de ce mécanisme. Finalement, les attributs contextuels trouvés dans une règle de configuration du mécanisme M « A_M^{ctx} » sont stockés dans la place « memory ». Cette mémoire, par la suite, peut être utilisée pour représenter des mécanismes à état (stateful machines). En outre, le lieu « memory » peut être partagé par différents mécanismes génériques.

Nous avons appliqué notre approche de modélisation sur différentes études de cas : utilisation de IPsec et du NAPT et l'analyse fine d'une configuration iptables. Nous en dégageons des éléments d'appréciation de la capacité d'expression et d'analyse de notre approche.

Les résultats sont encourageants car les conflits ont été détectés sans nécessiter de connaissances ou d'expérience a priori dans le domaine. Ceci nous pousse à croire que notre approche peut être utilisée pour détecter des conflits non connus impliquant de nouveaux mécanismes de sécurité pouvant agir à différents niveaux des piles de protocoles.

1.3 Organisation de la thèse

Suite à cette introduction, cette thèse expose dans le Chapitre 2 les aspects organisationnels, les aspects normatifs et les principales solutions de sécurisation. En outre, ce chapitre permet d'approcher la complexité de la tâche de sécurisation et de positionner la contribution de cette thèse en réponse à des insuffisances de gestion des systèmes informatiques interconnectés.

Plus précisément, la sécurisation d'un système ne saurait s'arrêter à une prise de conscience. Pour cela des solutions ont été proposées pour évaluer la résistance des systèmes, des infrastructures et des produits des technologies de l'information que nous illustrons dans la section 2.2. Ces solutions se basent sur un référentiel issu de différents travaux convergents, de l'approche initiatrice américaine des TCSEC, aux critères communs (CC) fédérant des travaux internationaux, en passant par les

ITSEC (cadre européen). Les critères, rigoureusement définis, permettent d'évaluer sur une échelle (EAL) les capacités d'un produit et la qualité de son cycle de vie. Dans la section 2.3, des travaux de normalisation sont rappelés et l'analyse montre tout l'intérêt de bon nombre de mécanismes de sécurité dans la construction d'une solution sécurisée d'une organisation qui doit répondre à des besoins opérationnels et réglementaires. Cette construction est souvent complexe dans la mesure où elle repose sur un groupement de solutions techniques listées dans la section 2.4. Cette dernière traite de la compréhension des principales solutions de sécurisation et ne peut s'arrêter aux strictes considérations protocolaires (celles des piles ISO/OSI, IETF ou encore ITU), ni s'arrêter à la prise en compte seule des problématiques de gouvernance des organisations. Enfin, nous terminons par dresser un inventaire des protocoles les plus réputés dans la sécurisation des échanges dans un environnement d'interconnexion de réseaux.

Dans la quête d'outils facilitant à l'administrateur la tâche de configuration du réseau et de sa sécurité, une littérature abondante pose le problème de l'inconsistance de mécanismes de sécurité entrant dans la construction de solutions sécurisées. Nous avons alors entrepris un état de l'art publié dans [El-Khoury et al. 2011 a] que nous avons repris dans le Chapitre 3. Le problème d'inconsistance de mécanismes de sécurité survient lorsque deux règles/configurations de sécurité sont contradictoires ou en conflit.

Une réponse à ce type de problème doit être formellement prouvée. Les études rencontrées ne nous ont pas permis de faire ressortir une solution de spécification formelle suffisamment pertinente. Par contre, nous avons pu mettre en exergue un besoin de modélisation formelle orientée flux pour l'analyse de politiques de sécurité réseau. Notre objectif vise la mise à disposition d'un langage de modélisation pour décrire et valider les architectures solutions répondant à des exigences de sécurité réseau.

Le 0Chapitre 4 est consacré à la définition d'un langage de modélisation formelle répondant à ce besoin. Cette contribution a fait l'objet de 2 publications [El-Khoury et al. 2011 b] [El-Khoury et al. 2012 a]. Le langage est indépendant des technologies rencontrées dans les solutions combinant les différents mécanismes de sécurité requis. Il permet en outre de décrire les capacités de traitement des flux (blocage, relais, filtrage, transformation/chiffrement à des fins d'authentification et/ou de confidentialité etc.) (Section 4.3) et la configuration des mécanismes de sécurité devant être mis en œuvre en construisant une abstraction des flux physiques de blocs de données (Section 4.4).

Afin d'évaluer les pouvoirs d'expression et d'analyse du langage, mais aussi et surtout de montrer les facilités de composition et de réutilisation inhérentes à la construction de solutions mettant en œuvre plusieurs mécanismes, nous avons alors exprimé dans la section 4.5 notre modèle dans le langage des réseaux de Petri colorés hiérarchiques. Nous avons alors bâti un modèle générique que nous avons appelé GAM pour Generic Attribute-based Mechanism model, comme composant logique de base représentant un mécanisme de sécurité. En utilisant le logiciel « CPN Tools », nous

pouvons alors analyser pas à pas l'impact des fonctions de transformation associées à un mécanisme ou à une combinaison de mécanismes sur les attributs des flux considérés.

Le Chapitre 5 présente des études de cas dans l'objectif de valider le modèle et vérifier notre approche sur des cas connus des administrateurs (par exemple, IPsec, NAPT et iptables). Dans la section 5.2, la spécification de notre formalisme a été présentée pour les protocoles AH et ESP en donnant un exemple de mise en œuvre de AH en mode tunnel. Il en est de même pour NAT et Netfiter/iptables dans les sections 5.3 et 5.4. Puis dans la section 5.5, quatre scenarii permettent d'illustrer en utilisant la représentation des flux de données, les attributs associés ainsi que les propriétés de sécurité recherchées en vue de pouvoir automatiser la détection de conflits lors de la composition des opérations de transformations. Ce travail a fait l'objet des publications [El-Khoury et al. 2012 b] [El-Khoury et al. 2013].

Chapitre 2. Approches de la sécurité dans le monde du numérique

Depuis plusieurs années la révolution numérique n'a cessé de gagner une place toujours plus importante modifiant les organisations, les méthodes de travail et les relations humaines dans les secteurs industriels, commerciaux et administratifs ; la révolution numérique a également impacté d'autres domaines que ceux de la simple productivité comme celui de la domotique, du domaine véhiculaire et plus généralement ceux de la vie privée. Aujourd'hui, cette expansion du numérique ouvre le champ au développement de nouvelles applications dans le cadre de l'Internet des objets. Cette révolution s'accompagne d'une prise de conscience de plus en plus forte, sur la nécessité de considérer la problématique de la sécurisation afin de construire et d'assurer la protection des systèmes matériels, des systèmes d'information, des réseaux, des données, de la propriété intellectuelle ou encore de la vie privée.

Ce chapitre a pour objectif de dresser un panorama, sans pour autant être exhaustif, permettant d'approcher la complexité de la tâche de sécurisation et de positionner la contribution de cette thèse.

2.1 Définition de la sécurité

Dans son sens le plus général, la sécurité est définie comme une situation dans laquelle quelqu'un, quelque chose (bien mobilier, immobilier) n'est exposé à aucun danger, à aucun risque, en particulier d'agression, d'accidents, de vol, ou de détérioration. Certains peuvent considérer que cette définition relève de l'idéal et peuvent y apporter des nuances. En particulier la sécurité caractérise une situation où les risques et dangers potentiels ont été analysés. Un système pourra être caractérisé de sécurisé dès l'instant où il présente les propriétés suffisantes pour résister aux sollicitations caractérisant les risques envisagés par le concepteur d'une solution.

Le numérique a conquis presque tous les domaines ; les facilités qu'il offre en matière de stockage et de traitement de l'information n'est plus à citer. La gestion de l'information a donné naissance à ses systèmes économiques. L'information est une source de convoitise de par les possibilités qu'elle offre. Comme toute richesse, l'information et les moyens techniques qui permettent d'y accéder ont donc besoin d'être protégées. Cependant la sécurité du numérique est une

problématique complexe. Elle est multi-facettes dans la mesure où il faut répondre à des attentes différentes, à des exigences qui parfois peuvent être contraires les unes aux autres.

Elle est liée d'une part à des problématiques organisationnelles. Ainsi les problématiques de sécurité concernent et impactent l'organisation interne des entreprises (cf. section 2.2). Les échanges qui se nouent entre des clients et des fournisseurs donnent naissance à des attentes du numérique pour que la confiance puisse se construire. La problématique de la sécurité nécessite que des solutions d'audit, de surveillance, de contrôle soient déployées pour apporter des garanties ou encore des niveaux d'assurance sur l'état d'un système d'information. En effet, la pérennité des organisations dépend davantage chaque jour de la fiabilité de son système d'information et il est donc incontournable de prendre en compte la sécurité au jour le jour.

D'autre part la construction d'une solution sécurisée passe par la prise en compte des travaux de normalisation (cf. section 2.3). Elle doit aussi prendre en considération l'existence de solutions opérationnelles qui dépassent le cadre dans lequel les normes sont pensées et proposées. La construction d'une solution sécurisée est le plus souvent d'ordre complexe, dans la mesure où elle repose sur une agrégation de solutions techniques présentant des fonctionnalités diverses dont la compatibilité n'est pas toujours (pour ne pas écrire rarement) assurée (cf. section 2.4).

La sécurité concerne d'une part les systèmes autonomes, des systèmes de stockage, des systèmes de traitement, les infrastructures de communication (réseaux, protocoles), les services logiciels mis en œuvre qu'ils soient locaux à un système ou encore accessibles à travers un réseau de communication. Chaque système, chaque infrastructure, chaque service doit donner lieu à une évaluation spécifique dans un but de sécurisation. Cependant il faut également qu'ils soient analysés dans un contexte global. En effet des équipements des infrastructures déclarés sécurisés en regard d'une politique qui a été définie, peuvent donner lieu à l'existence d'incohérences sur un plan global. Des configurations inadaptées peuvent apparaitre et amener de l'inconsistance dans le déploiement et la mise en œuvre de solutions opérationnelles. La contribution apportée dans cette thèse permet en particulier de s'assurer de la compatibilité ou de l'incompatibilité qui résulte de l'assemblage de fonctionnalités associées à des équipements réseaux.

2.2 Aspects organisationnels

2.2.1 Attentes en matière de sécurité

Très tôt des tentatives ont été faites pour tenter de caractériser la sécurité et notamment dans le monde du numérique. Dans son ouvrage Computer Security Handbook [NIST95], le NIST a formalisé ce que recouvrait le terme sécurité dans le contexte de l'informatique. Il l'a défini comme « *La protection accordée à un système d'information automatisé afin d'atteindre les objectifs applicables à la préservation de l'intégrité, la disponibilité et la confidentialité des ressources du système*

d'information (y compris le matériel, les logiciels, le firmware, les informations/données) et des télécommunications ».

Dans cette appréciation de la sécurité :

- L'intégrité des données qui se réfère à la protection contre le changement ou l'altération: les données transmises par un émetteur sont-elles identiques à celles reçues par la cible ?
- La disponibilité des données qui se réfère à la protection contre les interruptions de service: les données aux usages légitimes sont-elles accessibles à tout moment ?
- La confidentialité des données qui se réfère à la protection contre l'accès non autorisé (e.g., via snooping or wiretapping).

Aujourd'hui ces propriétés restent toujours d'actualité et s'appliquent au monde du numérique et à tous ses aspects qu'ils relèvent du niveau organisationnel ou encore du niveau technique.

Pour d'autres la sécurité rejoint la sûreté. Ainsi, selon [Laprie et al.1995], [Avizienis et al 2004], la sûreté de fonctionnement d'un système informatique se définit comme « l'aptitude à délivrer un service avec un niveau de confiance justifié ». Cette définition permet de conclure qu'un système doit être caractérisé par sa faculté à répondre à des exigences et que son comportement doit donner lieu à la délivrance de toute appréciation sur le comportement du système.

La sûreté de fonctionnement d'un système peut être établie dès lors que les attributs qui la caractérise sont donnés, que les entraves qui nuisent à sa réalisation sont identifiés et dès lors que les moyens qui permettent de garantir cette sûreté sont inventoriés. Les promoteurs de la sûreté de fonctionnement considèrent des nuances en introduisant un attribut de fiabilité qui caractérise la faculté à garantir la continuité d'un service, la sécurité innocuité qui caractérise l'absence de conséquences sur l'environnement, la maintenabilité qui se réfère à la possibilité de réparer un système voire de le faire évoluer [Lastera 2014].

Les entraves à la sûreté de fonctionnement correspondent aux circonstances indésirables sans que pour autant elles soient inattendues. Les entraves portent atteinte à la sûreté de fonctionnement : les défaillances, les erreurs et les fautes. Une défaillance survient lorsque le service délivré par le système s'éloigne de l'accomplissement de la fonction du système. Une erreur est la partie de l'état du système susceptible d'entraîner une défaillance. Une faute est la cause adjugée ou supposée d'une erreur.

Ces moyens sont les méthodes et techniques qui permettent d'atteindre un comportement souhaité. Les moyens employés ont entre autres pour objectifs de répondre à des besoins tels que la prévention, la tolérance, la surveillance, la maintenance corrective. Les mesures de prévention tentent d'empêcher l'occurrence d'une défaillance, la tolérance s'attache à fournir un service en dépit des

fautes, la surveillance estimer la présence, la création et les conséquences des fautes, la maintenance corrective a pour objectif de réduire la présence (nombre, sévérité) des fautes.

La sécurisation d'un système ne saurait s'arrêter à une prise de conscience. Il est bien sûr nécessaire qu'au-delà de la mise en avant des attributs de sécurité et de sûreté, il y ait une des métrologies qui soient proposées pour évaluer la résistance des systèmes, des infrastructures, des produits des technologies de l'information (TI).

2.2.2 Apports de l'approche TCSEC

Dès que les premières infrastructures réseau ont été déployées les premières questions se sont posées. Les TCSEC (Trusted Computer System Evaluation Criteria) [TCSEC 1985] sont un ensemble de critères retenus pour évaluer la fiabilité de systèmes informatiques centralisés. A l'origine cette norme américaine a été initiée par le DOD (Department Of Defense) des Etats-Unis. Elle a défini au sein de l'Orange Book en décembre 1985, puis au sein du Red Book en juillet 1987 des règles de sécurité permettant de déterminer le degré de protection d'un système d'exploitation avec au total 7 niveaux croissant de sécurité. Dans cette classification (Tableau 1) cinq groupes ont été définis avec des niveaux notés par un nombre dont la valeur est d'autant plus importante que la protection augmente.

Tableau 1. Degré de protection d'un système d'exploitation

Niveau	Evaluation protection	Commentaire
D	Protection minimale	Systèmes qui n'ont pas pu satisfaire aux exigences de classe de niveau supérieur
C1	Protection discrétionnaire	Cette classe ne suppose pas fondamentalement de différences d'habilitation parmi les usagers. Les caractéristiques de systèmes de cette classe tendent en fait à empêcher les utilisateurs de commettre des erreurs qui pourraient nuire au système ou aux autres utilisateurs.
C2	Protection discrétionnaire contrôlée C1 < C2	En plus du niveau C1, un système classé C2 doit fournir : - Une surveillance des utilisateurs (fichiers de logs). - Contrôle d'accès au niveau utilisateur (et non groupe). - L'assurance qu'aucune donnée laissée en mémoire dans le système ne peut devenir accidentellement accessible à un autre utilisateur.
B	Protection mandataire B1<B2<B3	Les systèmes de la classe B1 doivent proposer au moins une relation entre une ressource et un utilisateur (étiquetage des fichiers et des utilisateurs pour fournir des habilitations).

		Dans la sémantique sujet/objet, un sujet est autorisé à lire un objet si son niveau d'habilitation est supérieur ou égal à la classification de l'objet. Un sujet est autorisé à écrire un objet si son niveau d'habilitation est inférieur ou égal à la classification de l'objet. Le niveau B2 prend en compte les possibilités d'atteinte non souhaité d'un système. En conséquence dans le livre orange de départ le niveau B2 associe la notion de résistance du système à des accès non souhaités. Le niveau B3 est atteint lorsqu'un administrateur et qui endosse donc la responsabilité est nommé.
A1	Protection vérifiée	Une seule classe a été définie. Ce niveau s'intéresse ici à l'existence d'une preuve mathématique sur la conformité du système aux spécifications en matière de sécurité. Les modèles formels doivent être utilisés.

Comme suite à une décision du Conseil des Communautés Européennes (31 mars 1992) en matière de sécurité des Systèmes d'information, l'engagement communautaire s'est concrétisé par des propositions visant à déployer une stratégie globale pour assurer la sécurité des systèmes d'information et de communication. Le plan d'action qui s'est dégagé a conduit au développement d'un cadre stratégique pour la sécurité des systèmes d'information, incluant les attentes des utilisateurs, des fournisseurs, et des prestataires de services ainsi que l'élaboration de spécifications de normes, d'évaluations, de certifications pour que la sécurité des systèmes d'informations puisse être effective.

2.2.3 Apports de l'approche ITSEC

Les ITSEC (Information Technology Security Evaluation Criteria) sont le résultat de cette volonté. L'harmonisation des travaux a concerné principalement les quatre pays européens Allemagne, France, Pays-Bas et Royaume-Uni.

Les ITSEC [ITSEC 1991] considèrent qu'une politique de sécurité est « *l'ensemble des lois, règles et pratiques qui régissent la façon dont l'information sensible et les autres ressources sont gérées, protégées et distribuées à l'intérieur d'un système spécifique* ». La différence essentielle entre les approches TCSEC et ITSEC vient de la distinction entre fonctionnalité et assurance. Une classe de fonctionnalité décrit les fonctions que doit mettre en œuvre un système tandis qu'une classe d'assurance décrit l'ensemble des preuves qu'un système doit apporter pour montrer qu'il implémente les fonctions qu'il prétend fournir. Les ITSEC proposent plusieurs classes de fonctionnalités de base [Fisch et White 1999] :

- les classes de fonctionnalité F1 ou F-C1, F2 ou F-C2, F3 ou F-B1,F4 ou F-B2, F5 ou F-B3 sont des classes de confidentialité qui correspondent aux exigences de fonctionnalité des classes C1 à A1 dans les TCSEC
- F6 : la classe de fonctionnalité F-IN concerne les cibles d'évaluation (TOE : Target Of Evaluation) pour lesquelles il y a des exigences d'intégrité pour les données et les programmes
- F7 : la classe de fonctionnalité F-AV impose des exigences de disponibilité
- F8 : la classe de fonctionnalité F-DI impose des exigences élevées pour l'intégrité des données au cours de leur transmission
- F9 : la classe de fonctionnalité F-DI impose des exigences élevées pour la confidentialité des données au cours de leur transmission
- F10 : la classe de fonctionnalité F-DX est destinée aux réseaux exigeants en matière de confidentialité et d'intégrité de l'information.

Au-delà de la définition des classes de fonctionnalité, les promoteurs des ITSEC ont introduit la notion de cible d'évaluation (TOE). Une TOE rassemble les différents éléments du contexte d'évaluation tels que le contenu de la politique de sécurité, les spécifications des fonctions attendues de la sécurité, le descriptif des mécanismes de sécurité (optionnel), les éléments permettant d'apprécier la résistance minimale des mécanismes mis en œuvre etc. Sept niveaux d'évaluation correspondant à des degrés de confiance dans la conformité d'une TOE ont été caractérisés dans le Tableau 2 suivant :

Tableau 2. Les critères d'évaluation extraits de [ITSEC 1991]

Niveau	Caractérisation
E0	Ce niveau représente une assurance insuffisante
E1	A ce niveau, il doit exister une cible de sécurité et une description informelle de la conception générale de la TOE. Les tests fonctionnels doivent indiquer que la TOE satisfait à sa cible de sécurité.
E2	Outre les exigences du niveau E1, il doit exister une description informelle de la conception détaillée. Les éléments de preuve des tests fonctionnels doivent être évalués. Il doit exister un système de gestion de configuration et un processus approuvé de diffusion.
E3	En plus des exigences du niveau E2, le code source et/ou les schémas descriptifs des matériels correspondants aux mécanismes de sécurité doivent être évalués. Les éléments de preuve des tests de ces mécanismes doivent être évalués.

E4	En plus des exigences du niveau E3, il doit exister un modèle formel sous-jacent de politique de sécurité supportant la cible d'évaluation. Les fonctions dédiées à la sécurité, la conception générale et la conception détaillée doivent être spécifiées en style semi-formel.
E5	En plus des exigences du niveau E4, il doit exister une correspondance étroite entre la conception détaillée et le code source et/ou les schémas descriptifs des matériels.
E6	En plus des exigences du niveau E5, les fonctions dédiées à la sécurité ainsi que la conception générale doivent être spécifiées en style formel de manière cohérente avec le modèle formel sous-jacent de politique de sécurité.

2.2.4 Apports des Critères Communs

Les CC (Critères Communs) [CC 1999 a et b] représentent l'aboutissement de la volonté de fédérer les travaux menés par différentes institutions et organismes pour l'évaluation de la sécurité. A la suite de l'initiative menée dans le cadre des TCSEC, divers pays ont entrepris de définir (ou enrichir) des critères d'évaluation basés sur les concepts des TCSEC mais plus flexibles et plus adaptables au caractère évolutif des TI.

Comme nous l'avons annoncé, le déploiement du numérique est tel aujourd'hui que beaucoup de biens se présentent sous la forme d'informations qui sont stockées, traitées et transmises par des produits ou systèmes des technologies de l'information. Des exigences en regard de ces informations numériques peuvent être formulées par propriétaires. En particulier les demandes peuvent être relatives à un contrôle sur la diffusion, la modification et l'exploitation. La confiance dans la sécurité des TI peut être obtenue par des actions menées à différents stades (processus de développement, d'évaluation et d'exploitation).

Ayant donné lieu à la norme [ISO 15408], les critères communs fournissent un guide pour le développement et le contrôle de produits des TI. Les produits des TI nécessitent de fait la mise en œuvre de systèmes d'exploitation, de réseaux, d'équipements spécifiques dans le cadre de la construction d'applications distribuées coopératives. L'apport de garanties nécessite donc que tout système soit évalué en fonction de l'usage qui doit légitimement en être fait. Comme dans le cadre TCSEC et ITSEC, les critères communs s'adressent aux clients et fournisseurs des TI. Ils s'adressent également aux auditeurs chargés de mener des évaluations.

Les critères communs ont apporté une classification des exigences fonctionnelles de sécurité. Chaque classe est désignée par un nom unique dont l'abréviation est constituée de trois caractères « Fxx » : « F » pour classe d'exigence fonctionnelle et « xx » pour identifier le nom de la fonctionnalité couverte : FAU (Audit de sécurité), FCO (Communication), FCS (Support cryptographique), FDP (Protection des données de l'utilisateur), FIA (Identification et Authentification), FMT (Administration de la sécurité), FPR (Protection de la vie privée), FPT

(Protection de l'ensemble des fonctions de sécurité), FRU (Utilisation des ressources), FTA (Accès à la TOE), FTP (Chemins et canaux de confiance). Cette liste peut être amenée à être complétée.

Les critères communs ont également apporté une classification des exigences d'assurance. La fiabilité d'un SI repose sur l'hypothèse qu'il n'y aura pas de défaillance. Il ne s'agit pas simplement de l'énoncer, encore faut-il se donner des moyens d'apporter des garanties et donc introduire des éléments de vérification dans le cadre de la production d'une technologie de l'information. Ils sont identifiés sous la forme Axx ou les xx caractérisent un domaine particulier : gestion de configuration, livraison et exploitation, etc. La liste est donnée ci-après. ACM (Gestion de configuration), ADO (Livraison et exploitation), ADV (Développement), AGD (Guides), ALC (Support au cycle de vie), ATE (Tests), AVA (Estimation des vulnérabilités), APE (Evaluation d'un profil de protection), ASE (Evaluation d'une cible de sécurité), AMA (Maintenance de l'assurance).

Une évaluation selon les Critères Communs aboutit à la remise d'un certificat d'évaluation qui certifie que le SI évalué respecte un certain niveau d'assurance. Ce niveau d'assurance est une note allant de la plus petite EAL1 à la plus forte EAL7 (Tableau 3).

Tableau 3. Compatibilité rétroactive de CC

EAL	Niveau de sécurité	TCSEC	ITSEC	
			Evaluation	**Fonctionnalité**
	Protection minimale	D	E0	--
EAL1	Test fonctionnels	--	--	--
EAL2	Test structurels	C1	E1	F1
EAL3	Test et vérification méthodiques	C2	E2	F2
EAL4	Conception, tests et vérification méthodiques	B1	E3	F3
EAL5	Conception semi-formelle et tests	B2	E4	F4
EAL6	Vérification semi-formelle de la conception générale	B3	E5	F5
EAL7	Vérification formelle de la conception générale	A1	E6	F5

Chaque niveau d'évaluation inclut l'intégralité des niveaux précédents. A partir du niveau EAL5 l'utilisation de méthodes semi-formelles ou formelles est nécessaire dans la mesure où des évaluations poussées doivent permettre de déterminer l'adéquation ou non à des exigences de plus en plus strictes. A titre d'exemple, B3 et A1 est le niveau d'assurance qui exige la reprise de confiance. B2 et B3 est le niveau d'assurance qui exige que la gestion de la configuration est appliquée au cours du développement et de la maintenance du système. Rarement qu'un dispositif ou un système peut être certifié à un niveau plus élevé à EAL5. Il faut noter que dans ce contexte le travail que nous proposons est à même d'enrichir la conception d'une solution.

2.3 Aspects normatifs

2.3.1 Sécurité pour les systèmes ouverts ISO 7498 et ITU- X.800

L'adoption du modèle ISO/OSI [ISO 7498-1] a eu comme effet d'harmoniser et de simplifier les difficultés rencontrées lors de l'interconnexion de systèmes. Ce n'est cependant qu'au travers de la publication des normes [ISO 7498-2] et de la recommandation [ITU-T, X.800] que les aspects sécurité ont retenu l'attention des acteurs de la normalisation des architectures réseau. Les références [RFC 2828] et [RFC 4949] ont également contribué à donner un glossaire et des définitions. Les documents en question fournissent les éléments généraux d'architecture liés à la sécurité qui peuvent être appliqués lorsque la protection est nécessaire. Une description générale de services de sécurité et des mécanismes associés y est donnée comprenant également des propositions d'implémentation pour les services de sécurité.

Par exemple dans [RFC 4949], les définitions données permettent également d'apprécier la pluralité de la construction de la sécurité.

Information system : est défini comme un ensemble organisé de ressources, de procédures informatiques et de télécommunications (incluant équipements et services qui, ensemble, permettent le support d'infrastructures, de méthodes et de personnes) qui permettent de créer, collecter, enregistrer, traiter, stocker, retrouver, afficher, contrôler ou de disposer d'informations afin d'accomplir une fonction spécifique.

Information security : recouvre les mesures qui permettent d'implémenter et d'assurer la sécurité des systèmes d'information. Or pour assurer cette sécurité, il faut effectivement s'attacher à protéger tous les éléments qui peuvent présenter des risques, des vulnérabilités et/ou être l'objet d'attaques. Les systèmes informatiques, les terminaux sous toutes leurs formes support de l'exécution d'application, les réseaux et leurs éléments constitutifs, les services (téléphonie, transmission de données, systèmes de stockage, application spécifiques, services web, etc.) doivent donc faire l'objet de protection.

Les attentes en matière de sécurité concernent la disponibilité, la confidentialité, et l'intégrité. Des mécanismes de sécurité sont étudiés, développés et déployés. Des services de sécurité peuvent naître de l'exploitation de plusieurs mécanismes de sécurité. La justification de l'implémentation de mécanismes de sécurité vient d'une part de la fragilité supposée ou avérée qui pèse sur des équipements, des infrastructures, des applications et des services mis en œuvre dans une solution numérique. L'existence d'une probabilité (non nulle) de menaces qui peuvent (à juste titre) entraîner des attaques et les conséquences financières, sociétales, humaines qui en découlent font le reste.

Les menaces peuvent être *accidentelles* ou *délibérées* et peuvent être *actives* ou *passives* [ITU-T, X.800]. Une menace accidentelle est caractérisée par la non-préméditation (par exemple dysfonctionnement non attendu ni prévu d'un système matériel ou logiciel). Une menace délibérée est

une menace dont la mise à exécution est calculée (lorsqu'une menace délibérée est mise à exécution, on parle alors d'*attaque*.) Une menace active est une menace résultant d'une modification d'état (par exemple altération de données ou destruction d'un équipement physique). Pour une menace passive, il n'y a pas de modification d'état (par exemple, l'écoute clandestine).

La découverte des fragilités est intimement liée à chaque élément. Dans le domaine du génie logiciel et en particulier du développement d'application, la sécurisation va faire en sorte qu'un programme en cours d'exécution ne se bloque pas, par exemple, pour cause d'insuffisance de mémoire. Dans le cadre de l'accès à des données, il faut s'assurer qu'un système d'autorisation soit mis en œuvre si l'on souhaite que l'accès à des fichiers ou base de données soit protégé. Dans le cadre des réseaux de communication, il ne faut pas que le chiffrement d'un datagramme soit réalisé alors que l'accès à des numéros de port, ou de champs mutables doit être réalisé. Les protocoles de communication donnent lieu à la publication de spécifications dans un cadre normatif. Toute étude de ces spécifications permet de suivre pas à pas son exécution sur un système et de découvrir si on le souhaite des éléments qui peuvent porter atteinte à la disponibilité, l'intégrité et la confidentialité.

2.3.1.1 Attaques et conséquences

Le terme attaque recouvre toute action qui vient compromettre la sécurité des informations appartenant à une organisation. Une attaque passive tente d'apprendre ou d'utiliser des informations du système, mais sans affecter immédiatement et directement ses ressources. Un exemple classique est donné par la capture puis l'analyse d'un trafic transitant sur un réseau de communication. Une attaque active quant à elle tente de modifier les ressources du système ou d'affecter leur fonctionnement. Dans le cadre des réseaux de communication de telles attaques sont données par la mascarade, le rejeu, la modification, le déni de service.

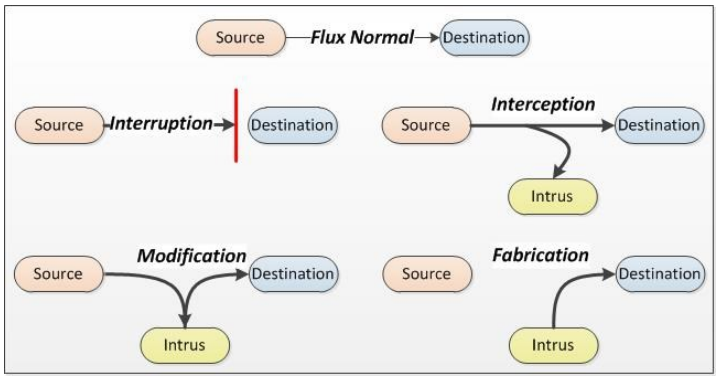

Figure 3. Les types d'attaques

Sous une forme générique, les effets de ces attaques entraînent l'interruption de services, l'interception de données, la falsification par modification de données ou encore la falsification par fabrication de fausses informations [William Stalling 2011] :

L'interruption se réfère à une situation dans laquelle les services ou les données deviennent indisponibles, inutilisables ou encore sont détruites. L'effet recherché met en cause l'exigence de ***disponibilité*** des informations et services attendus.

L'interception se réfère à la situation où un accès non souhaité, non autorisé est réalisé. Les attaques concernées portent sur l'exigence de ***confidentialité*** des informations. Un exemple typique d'interception est l'écoute par une tierce partie d'une communication qui se déroule entre deux entités.

La falsification par modification implique une modification non prévue et non autorisée qui affecte des données ou la réalisation d'un service. De telles attaques compromettent l'exigence d'***intégrité*** des données ou de la réalisation d'un service. La falsification des entrées d'une base de données, la modification d'un programme pour qu'il enregistre secrètement les activités de son utilisateur en sont des exemples.

La falsification par fabrication se réfère à la situation dans laquelle des données ou des demandes d'exécution de services sont anormalement générées. Cette attaque porte atteinte à une exigence autre qui est celle de l'***authenticité*** qui peut être associée à un système ou un acteur d'un système (personne humaine, programme). Par exemple, un intrus peut tenter d'ajouter de nouvelles entrées dans un fichier ou une base de données contenant des mots de passe.

2.3.1.2 Services de sécurité

Initialement les services de sécurité ont été décrits dans les documents [ISO7498-2] et [ITU-T, X.800]. Ils ont été regroupés en cinq catégories et quatorze services spécifiques. La vision est restée orientée très protocole de communication dans la mesure où l'idée était d'apporter des solutions de sécurisation des protocoles mis en œuvre. Les cinq catégories de base qui ont été proposées sont *le contrôle d'accès, l'authentification, la confidentialité des données, l'intégrité des données et la non-répudiation*. A titre d'exemple si l'on regarde le service d'authentification il est décrit comme : *«l'authentification d'une entité homologue communicante et l'authentification de la source des données».* Lorsque ce service est fourni par la couche (N), il confirme à l'entité (N+1) que l'entité homologue est bien l'entité (N+1) déclarée. Ce service est prévu pour être utilisé lors de l'établissement de la phase de transfert de données d'une connexion, ou parfois pendant cette phase, pour confirmer les identités d'une ou de plusieurs entités connectées à une ou plusieurs autres entités. Ce service garantit (uniquement lors de son utilisation) qu'une entité n'essaie pas d'usurper une identité ou de répéter une ancienne connexion de façon non autorisée. Des schémas d'authentification unilatérale ou mutuelle d'entités homologues, avec ou sans contrôle réitéré, sont possibles et peuvent donner divers degrés de protection.

Les travaux de l'ITU-T ont donné lieu en 2003 à une opération de révision à travers la nouvelle recommandation [ITU-T, X.805]. Celle-ci prend en compte la possibilité de construire une architecture sécurisée de bout en bout et indépendante de la technologie sous-jacente du réseau. Les

principes généraux et les définitions ont été établis pour s'appliquer à toute application, bien que les menaces, les vulnérabilités, et les mesures visant à les contrer puissent changer d'une application à l'autre. Dans cette nouvelle approche trois services complémentaires de sécurité ont été amené. Ils traitent de la *sécurité de la communication*, de la *disponibilité* et celui du *respect de la vie privée*.

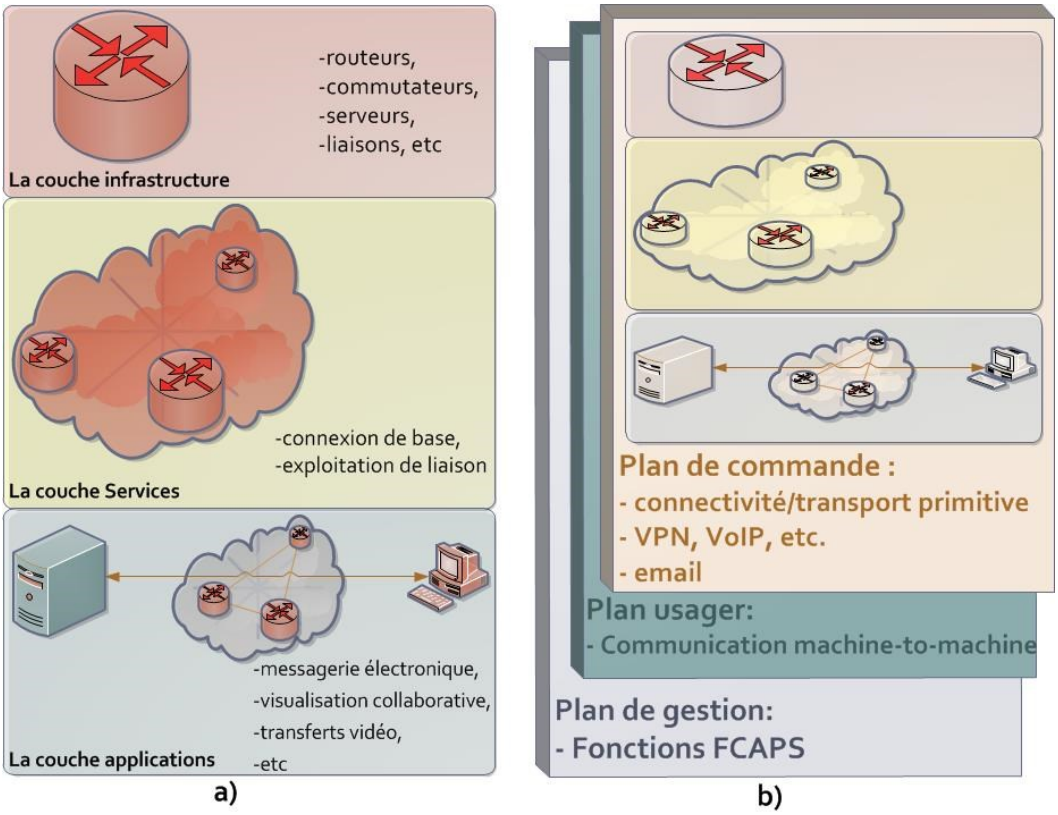

Figure 4. Les trois plans de sécurité (a) sur les trois couches de sécurité (b)

Des travaux de l'IUT T concernant les opérateurs de réseaux et télécommunication, on retrouve les concepts retenus dans le cadre de la modélisation de l'architecture RNIS large bande. Une organisation doit prendre en compte des besoins dépendant du plan de commande, du plan usager et du plan de gestion (Figure 4 b). Par ailleurs on y trouve une structuration où apparaissent trois couches clairement identifiées (Figure 4 a). La couche de sécurité relative à l'infrastructure comprend les installations de transmission de réseau et les différents éléments de réseau (routeurs, commutateurs, serveurs, liaisons, etc.) protégés par les mesures de sécurité. La couche de sécurité relative aux services de réseau offerts aux clients (connexion de base, exploitation de liaison, etc.). La couche de sécurité relative aux applications telles que la messagerie électronique, la visualisation collaborative, les transferts vidéos, le travail collaboratif, etc.

- Le contrôle d'accès permet d'assurer la protection contre toute utilisation non autorisée de ressources. Appliqué au réseau de communication, le contrôle d'accès permet de garantir que seuls les personnes ou les dispositifs autorisés peuvent accéder aux éléments de réseau, aux informations stockées, aux flux d'informations, aux services et aux applications.

- L'authentification sert à confirmer les identités des entités qui communiquent. Ce service garantit la validité des identités déclarées des entités en communication (personne, matériel, application). Par essence il donne l'assurance qu'une entité ne tente pas d'usurper l'identité d'une autre entité.
- L'intégrité des données permet de garantir que les données sont correctes ou exactes. Les données sont protégées contre toute modification, destruction, création et/ou reproduction non souhaitée et non autorisée.
- La non-répudiation empêche une entité (éventuellement humaine) de nier avoir exécuté une action particulière liée aux données, grâce à la fourniture de diverses preuves d'actions dans le réseau.
- La confidentialité des données permet de protéger les données contre toute divulgation non autorisée. La confidentialité des données permet de garantir que le contenu des données ne pourra pas être exploité par des entités non autorisées (les moyens sont par exemple le chiffrement, les listes de contrôle d'accès, et les permissions d'accès aux ressources etc.).
- La sécurité de la communication amène la garantie que les informations ne seront acheminées qu'entre les points d'extrémité autorisés (il faut donc des mécanismes adaptés pour prouver l'impossibilité d'interceptions ou de détournement).
- La disponibilité permet de garantir qu'il n'y a pas déni de l'accès autorisé aux éléments de réseau, aux sites accédés, à la transmission de flux d'informations, donc pas de déni de services.
- Le respect de la vie privée permet d'assurer la protection des informations qui pourraient être obtenues à partir de l'observation des activités dans le réseau (« tracking » des utilisateurs, de la navigation, collecte des adresses de machines etc.).

La construction de ces services repose sur l'utilisation et l'exploitation de mécanismes de sécurité

2.3.1.3 Les mécanismes de sécurité

Les mécanismes de sécurité ont également été définis dans les documents [ISO 7498-2] et [ITU-T, X.800]. Certains sont d'un caractère général alors que d'autres relèvent d'un usage plus spécifique. Ils sont résumés dans les tableaux ci-après (Tableau 4 et Tableau 5).

Tableau 4. Les mécanismes de sécurité spécifiques

Mécanismes de sécurité spécifiques	
Chiffrement	Basé sur l'exploitation de clés les algorithmes de chiffrement symétrique ou asymétriques assurent la transformation des données.

	Le niveau de sécurisation atteint dépend de la robustesse des clés. Le chiffrement peut assurer la confidentialité soit des données, soit du flux de données et peut jouer un rôle dans un certain nombre d'autres mécanismes de sécurité
Signature numérique	Consiste à l'ajout d'une marque (signature) pour vérifier l'intégrité ou l'origine des données
Contrôle d'accès	vérifie les droits d'accès d'un acteur aux données. Il n'empêche pas l'exploitation d'une vulnérabilité. (De nombreux mécanismes imposent des droits d'accès aux ressources
Intégrité des données	Une variété de mécanismes utilisés pour assurer l'intégrité d'une unité de données ou d'un flux d'unités de données
Echange d'authentification	Un mécanisme destiné à garantir l'identité d'une entité au moyen d'échange d'informations.
Mécanisme de bourrage	Données ajoutées pour assurer la confidentialité, notamment au niveau du volume du trafic.
Notarisation	L'utilisation d'un tiers de confiance pour assurer certains services de sécurité

Tableau 5. Les mécanismes de sécurité à caractère commun

Mécanismes de sécurité à caractère commun	
Fonctionnalité de confiance	Ce qui est perçu comme étant correct par rapport à certains critères (établis par une politique de sécurité)
Étiquette de sécurité (Security Label)	Marquage lié à une ressource (qui peut être une unité de données) désignant les attributs de sécurité de la ressource. (niveau de sensibilité, nature de la clé de chiffrage etc.)
Détection de l'événement	Détection des événements relevant de la sécurité – violations, événements normaux (demandes, accès réussis etc.)
Journalisation d'audit de sécurité	Les journaux d'audit de sécurité peuvent mener à la détection de violations de sécurité a posteriori. La collection des informations est en soi un mécanisme de sécurité
Reprise de sécurité	Traite des demandes provenant de mécanismes, tels que les fonctions de traitement et de gestion d'événements, et prend des actions de reprise immédiates ou différées.

[ITU-T, X.800] indique également de manière précise la relation entre quelques services de sécurité et des mécanismes de sécurité (cf. Tableau 6). D'une part, il est presque impossible d'avoir un « Y » dans toutes les cellules du tableau et d'autre part il est infaisable de combiner tous les mécanismes et les services pour compléter le tableau et avoir une sécurité idéale.

Tableau 6. Relation entre les services de sécurités et les mécanismes

mécanisme / Service	Chiffrement	Signature numérique	Contrôle d'accès	Intégrité des données	Echange	Bourrage	Contrôle de routage	Notarisation
Authentification de l'entité homologue	Y	Y			Y			
Authentification de l'origine des données	Y	Y						
Contrôle d'accès			Y					
Confidentialité	Y						Y	
Confidentialité du trafic des données	Y					Y	Y	
Intégrité des données	Y	Y		Y				
Non-répudiation		Y		Y				Y
Disponibilité				Y	Y			

2.3.2 Sécurité pour les systèmes ouverts ISO 7498 et ITU-T,X.800

La famille des normes de la série [ISO/IEC 27000] traite également de la sécurité de l'information. Les différents composants de cette série sont donnés dans le tableau ci-dessous. On y trouve différents aspects qui relèvent de la sécurité, et en particulier des informations relatives à la mise en œuvre d'un système de management de la sécurité de l'information (SMSI) [Kamel 2008]

- une vue d'ensemble de la famille de normes du SMSI (en fait de la famille ISO 2700x, Figure 5)
- une introduction aux systèmes de management de la sécurité de l'information (SMSI)
- une brève description du processus PDCA d'amélioration continue : Planifier-Déployer-Contrôler-Agir

- les principales définitions des termes utilisés dans la famille de normes du SMSI.

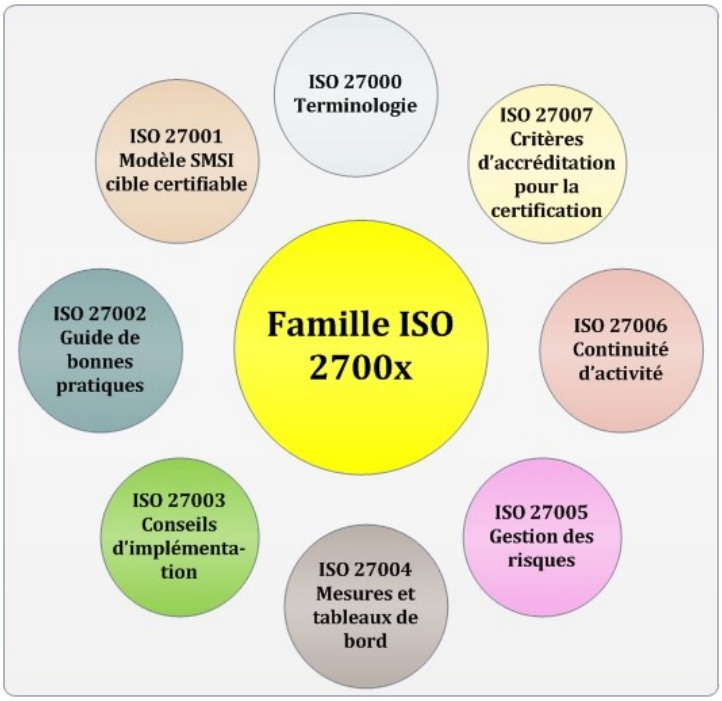

Figure 5. La famille ISO 2700x

La norme ISO/IEC 27001:2013 [ISO/IEC 27000] peut être considérée comme le cœur de cette famille. Elle définit les exigences en matière de mise en œuvre et de documentation du système de management de la sécurité de l'information. Cette norme apporte des éléments qui sont davantage du ressort de la gouvernance des entreprises, que de la sécurité vue sous un angle opérationnel et dans le domaine des réseaux de communications. Elle comporte en particulier des chapitres traitant :

- du contexte de l'organisation : périmètre du SMSI, interfaces, domaine d'application
- du leadership : engagement de la Direction Générale et définition des responsabilités vis à vis du SMSI
- de la planification : management des risques et définition du portefeuille des mesures de sécurité
- des ressources : ressources humaines et compétences, communication (interne & externe), gestion de la documentation (sécurité et SMSI)
- du fonctionnement : contrôles opérationnels, appréciation et traitement des risques
- de l'évaluation des performances : audit interne, revues de Direction, surveillance, mesures
- des améliorations : traitement des non-conformités, actions correctives, amélioration continue

Bien que cette norme traite également de la sécurité, son ambition dépasse largement le cadre du travail mené dans cette thèse. Il n'en demeure pas moins que notre contribution trouve une place dans ce cadre, puisque l'intérêt de notre travail est d'anticiper d'éventuelles erreurs de configuration ou de reconfiguration de systèmes en nous appuyant sur la modélisation des flux d'information. Nos travaux peuvent donc participer à peupler les indicateurs du SMSI en regard des exigences de sécurité qui peuvent figurer dans le document de politique de sécurité.

2.4 Principales solutions de sécurisation

Comme nous l'avons indiqué, les travaux de normalisation [ISO 7498-2], [ITU T, X.800] et [ITU T X.805], se sont inscrits dans la continuité de la normalisation des communications. Ils ont certes apporté des éléments relatifs à la sécurité, mais dans un état d'esprit principal qui était celui de la sécurisation du fonctionnement des protocoles de communication. Or la construction d'une architecture sécurisée ne peut pas s'arrêter aux strictes considérations protocolaires (celles des piles ISO/OSI, IETF ou encore ITU), ni s'arrêter à la prise en compte des problématiques de gouvernance des organisations.

Dans ce contexte, des solutions sécurisées concernant le monde du numérique ont été et sont encore déployées pour répondre à des besoins spécifiques. Certaines solutions considèrent davantage les problématiques et attentes des utilisateurs alors que d'autres donnent la priorité aux problématiques d'ordre architectural.

2.4.1 Gestion des identités

Le concept de gestion des accès et des identités (IAM en anglais) fait partie intégrante de la sécurité des organisations [Cheaito 2012]. La gestion des identités prend en considération la problématique de création, modification et destruction des comptes et privilèges associés à des utilisateurs. Elle permet également de s'assurer de l'authentification des utilisateurs, de garantir l'accès aux ressources et services en fonction de critères polymorphes. De plus, elle doit permettre d'historiser certains accès tout en garantissant la protection des libertés individuelles, et de traiter les autorisations. La gestion des identités est venue dans les années 80. La recommandation X.500 [ITU-T, X.500] a donné naissance à une gestion avancée des identités en ouvrant la voie à la constitution d'annuaires DAP (Directory Access Protocol).

2.4.2 Gestion des accès

Le concept de gestion des accès fait référence au contrôle et à la sécurisation des accès aux ressources ou services, et donc à la protection des informations contre tout accès non-autorisé. Le contrôle d'accès recouvre donc un sens particulier car il garantit la confidentialité des informations. Différents travaux ont été menés sur la mise en œuvre de systèmes de contrôle d'accès. Dans le travail de Sandhu et Samarati [Sandhu et Samarati 1994], le processus d'autorisation est présenté sous le

nom du « contrôle d'accès ». Dans les travaux de l'IETF [RFC 2989] la signification du « contrôle d'accès » recouvre trois processus nommés « authentification », « autorisation » et « audit ». En sécurité informatique, le « contrôle d'accès » se réfère à l'utilisation de mécanismes pour permettre aux entités authentifiées d'exécuter des actions en fonction de leur niveau d'autorisation et d'empêcher les entités de réaliser des actions non autorisées. Les droits des utilisateurs sur les fichiers (lire, écrire, exécuter), la constitution de listes de contrôle d'accès associés à des comptes utilisateurs, à des ressources, à des routeurs filtrants, firewalls (pare-feu en français), commutateurs en sont des exemples.

2.4.3 Système de gestion de clés

Dans les mesures de sécurité, l'exploitation des techniques de chiffrement propose de transformer des informations initialement codées sous une forme dans un nouveau format en utilisant une clé de chiffrement. La sécurité est donc apportée par cet usage. Actuellement, l'exploitation de clés publiques – chiffrement asymétrique – connait de nombreux champs d'application (commerce électronique, protection de contenus, signatures électroniques, etc.). La technologie de chiffrement réalise des opérations mathématiques sur des données numériques. En aucun cas les techniques de chiffrement n'offrent un ensemble complet pour la gestion et la distribution des clés publiques. C'est au sein de l'ICP (Infrastructures à Clés Publiques) (resp. PKI (Public Key Infrastructure)) qu'est assurée l'opération de gestion des clés. Cette architecture recouvre la génération, la distribution et la révocation des clés publiques. Dans cette infrastructure de gestion, l'autorité de certification a un rôle majeur puisqu'elle est la source de toute la garantie donnée aux utilisateurs et qui ont fait ce choix de dépendance vis à vis des services offerts.

2.4.4 Protocoles de sécurisation des échanges (SSL/TLS, SSH, IPsec)

Les systèmes de sécurisation des échanges ont pour vocation d'empêcher que des données transitant à travers une architecture non sécurisée ne puissent donner lieu à des interceptions. Plusieurs types de protocoles proposent de participer à cette protection des données. Les protocoles tels SSL/TLS agissent au plus près des applications alors que le protocole IPsec considère que la protection doit être assurée au niveau du réseau.

- ***SSL/TLS (Secure Socket Layer, Transport Layer Security)***

Développé par Netscape en 1995, le protocole SSL [RFC 6101] standardisé par l'IETF sous le nom TLS [RFC 5246] s'est rapidement imposé comme le mode de sécurisation privilégié des transmissions de données sur Internet. Intégré aux principaux navigateurs et serveurs Web, SSL utilise des techniques de chiffrement et un système de clés publique/privée initialement développé par RSA.

Le protocole SSL s'exécute au-dessus de TCP/IP, et en dessous des protocoles applicatifs tels que HTTP ou IMAP. Il supporte l'utilisation d'un grand nombre d'algorithmes de chiffrement, pour

les opérations d'authentification réciproque d'un serveur et d'un client, la transmission de certificat, ou encore l'établissement de clés de session. Le protocole SSL inclut deux sous-protocoles : le protocole SSL record et le protocole SSL « handshake » (négociation SSL). Le protocole SSL record définit le format utilisé pour la transmission des données. Le protocole de négociation SSL utilise le protocole SSL record pour échanger une série de messages entre un serveur et un client lorsqu'ils débutent une connexion SSL. Cet échange de messages est destiné à faciliter les actions suivantes :

- Authentifier le serveur auprès du client
- Permettre au client et au serveur de sélectionner un algorithme de chiffrement, supporté par les deux
- Éventuellement, authentifier le client auprès du serveur
- Utiliser des techniques de cryptographie à clef publique pour générer des éléments secrets partagés
- Établir une connexion SSL chiffrée

- ***SSH (Secure Shell)***

SSH [RFC 4253] désigne à la fois un programme et un protocole de connexion et d'exécution de commande sur un système accessible depuis un réseau. Le protocole sécurise la communication en utilisant un mécanisme de clé de chiffrement pour tous les paquets échangés. SSH est à la base un protocole qui a donné naissance à un ensemble de programmes. Il offre plusieurs fonctionnalités comme l'établissement de sessions sécurisées sur un système distant, le transfert de fichiers sécurisés ou encore l'établissement de tunnels sécurisés. SSH s'appuie également sur une architecture de type client/serveur pour assurer l'authentification des entités communicantes, la confidentialité et l'intégrité des données transmises sur un réseau. Il faut noter que SSH utilise deux algorithmes pour assurer l'authentification (RSA et DSA version 2 du protocole). Le chiffrement des données exploite les algorithmes de chiffrement DES, 3DES, IDEA, Blowfish pour la version 1 et AES, 3DES, Blowfish, Twofish, Arc-four ou encore Cast128 pour la version 2. Dans notre contribution, nous verrons que dans la modélisation proposée nous sommes à même de traiter l'existence de ces différences qui peuvent conduire à des inconsistances dans le cadre des configurations.

- ***IPsec (IP security)***

Alors que SSL/TLS, SSH sont des protocoles qui considèrent une sécurisation au plus près du niveau applicatif, IPsec (Internet Protocol Security) [RFC 4301], propose une sécurisation au cœur du réseau de communication. Défini par l'IETF il utilise les techniques de chiffrement pour protéger les informations contenues dans les datagrammes IP. IPsec opère donc au niveau de la couche réseau ce qui le rend indépendant des applications. Une présentation plus complète est donnée dans la partie expérimentation de cette thèse (cf. section 5.2) car nous serons amenés à présenter des inconsistances qui se produisent lorsque plusieurs mécanismes consécutifs sont appliqués sur des unités de données.

2.4.5 Interconnexion des réseaux des organisations

Les réseaux locaux d'entreprise (LAN) supportent des connexions entre systèmes en interne, mais également externes car les besoins de communication sont tels que l'accès au réseau Internet est requis pour satisfaire des connexions avec des filiales, des partenaires ou encore simplement pour accéder aux services du réseau Internet. Pour réaliser de telles opérations d'interconnexion, plusieurs architectures sont possibles :

- Les opérateurs télécoms sont à même de proposer des liaisons spécialisées pour assurer cette interconnexion. Pendant de nombreuses années les liaisons spécialisées, les solutions de type « groupe fermé » ont été exploitées par les entreprises. Dans le cadre de réseaux métropolitain cette interconnexion peut être réalisée en s'appuyant sur la mise en œuvre de réseaux locaux virtuels (VLAN) en exploitant les capacités des équipements à réaliser des commutations de trames par port. Dans cette situation cela revient à dédier une ligne physique à cette interconnexion
- Une autre solution consiste à exploiter le réseau Internet. L'avantage très clair de ce choix est donné par l'omniprésence des points de connexion à ce réseau, (accès à travers des liaisons ADSLn utilisation de la 3G, de la 4G, présence de « hotspots », réseaux WIFI supports, etc.). Aujourd'hui même si l'on peut regretter par moment le manque de débit, cette structure de réseau est largement rentrée dans les usages. La diversité des terminaux proposés témoigne de ce succès. Toutefois si l'on veut organiser des communications qui soient sécurisées deux options se présentent :
    - La première consiste à utiliser les mécanismes de tunnelisation dans le cadre de la construction de VPN (Figure 6). Que l'on soit dans la même entreprise ou que l'on dépende d'organisations différentes, un tunnel sécurisé doit être mis en place pour construire une liaison sécurisée entre les deux sites. C'est à travers le document de politique de sécurité que les choix en matière de sécurisation seront donnés. La construction de tunnels sécurisés peut faire appel à différents mécanismes, et différentes solutions pour assurer le chiffrement.

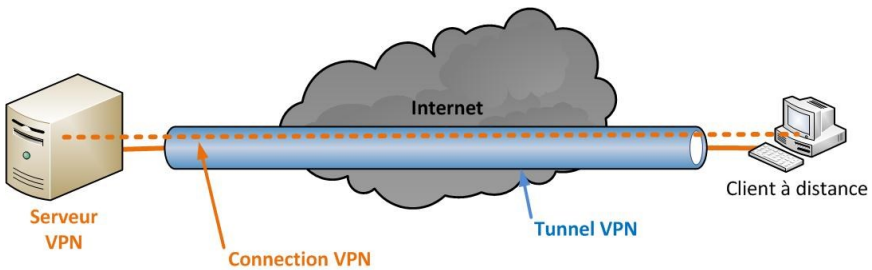

Figure 6. Lien VPN

- GRE (Generic Routing Encapsulation) [RFC 2784], consiste en l'ajout d'un en-tête qui contient les informations relatives au tunnel
- PPTP (Point-to-Point Tunneling Protocol) [RFC 2637], est un protocole de niveau 2 qui rend possible l'interconnexion des réseaux via un réseau IP. Il est développé par Microsoft, 3Com, Ascend, US Robotics et ECI Telematics. Permet d'établir des connexions PPP au travers de réseaux IP comme l'Internet. Basé sur deux composantes : TCP port 1723 pour la connexion de contrôle et de PPP dans IP via GRE pour l'encapsulation [RFC 2637]
- L2F (Layer Two Forwarding), est un protocole de niveau 2 développé par Cisco, Northern Telecom et Shiva
- L2TP (Layer Two Tunneling Protocol), protocole de niveau 2. Il est l'aboutissement des travaux de l'*IETF* [RFC 2661] pour faire converger les fonctionnalités de *PPTP* et *L2F*. Il encapsule de trames PPP dans L2TP
- IPSec [RFC 4301], est un protocole de niveau 3, issu des travaux de l'IETF, permettant de transporter des données chiffrées dans les réseaux IP. Il est principalement utilisé pour créer des tunnels fixes, particulièrement de routeur à routeur
- PPPoE (point-to-point protocol over Ethernet) [RFC 2516] : est un protocole d'encapsulation de PPP sur Ethernet. Il est employé par les connexions haut débit à Internet par ADSL et dans les réseaux « câbles » destinés aux particuliers, bien qu'une connexion utilisant un pont Ethernet-Ethernet soit souvent plus stable et plus performante

- La deuxième option est de réaliser un accès direct au réseau Internet et de mettre en place des composants dans la structure du réseau qui vont assurer une inspection des unités de données transmises dans la structure du réseau (Figure 7). Comme pour la situation précédente, le document de politique de sécurité doit exprimer les contraintes et les limitations qui doivent être appliquées sur les unités de données. Le document de politique doit donc définir comment protéger le plan d'adressage des systèmes, les services, les ressources, etc. Les firewalls, les systèmes de détection d'intrusion sont des exemples de système contribuant à la sécurisation dans cette situation.

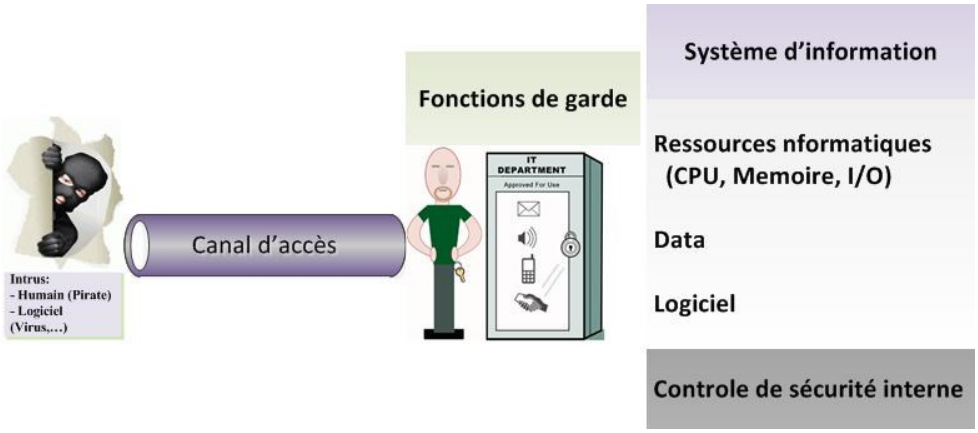

Figure 7. Système sécurisé

- un système de détection d'intrusion (IDS) [Bishop 2002] tente d'identifier les menaces dirigées contre le réseau de l'entreprise. Il s'appuie sur plusieurs sources d'informations comme les fichiers d'audit, les journaux de sécurité et le trafic réseau. Placés après les firewalls, les IDS constituent la dernière barrière de sécurité. Ils analysent le trafic qui passe à travers les firewalls et supervisent les activités des utilisateurs sur le réseau local. Par ailleurs, placés avant les firewalls, les IDS découvrent les attaques à l'entrée du réseau. Les IDS s'appuient généralement sur deux sources d'information : les paquets transitant sur le réseau et les informations collectées sur les machines.
- Un firewall est un système physique ou logique qui inspecte les paquets entrants et sortants du réseau afin d'autoriser ou d'interdire leur passage en se basant sur un ensemble de règles appelées ACL. Il existe principalement trois types de firewall :
  - Firewall avec filtrage des paquets
  - Firewall à filtrage des paquets avec mémoire d'états
  - Firewall proxy

2.5 Conclusion

Dans ce chapitre, nous avons proposé une synthèse de différents aspects portant sur la sécurité. En partant de la définition de la sécurité, nous avons rappelé que sa mise en œuvre peut être analysée à différents niveaux et donne lieu à des travaux menés par des acteurs différents.

Le niveau organisationnel impacte la gouvernance. A ce niveau les exigences de sécurité sont établies. Les méthodes permettant de capturer les exigences et en particulier celles relevant de la sécurité n'ont pas été traitées car elles ne font pas partie de ce travail. La nécessité de pouvoir apporter des indicateurs ou des preuves sur le respect des engagements a donné lieu tour à tour à la construction de référentiels. TCSEC, ITSEC, CC ont pour intérêt de pouvoir qualifier sur une échelle

de valeur des solutions qui sont proposées. Le positionnement sur une échelle de valeur implique qu'un audit soit réalisé. La conformité à des exigences de sécurité n'y échappe pas.

Les organismes de normalisation ont également œuvré. Cependant comme nous l'avons souligné, les mécanismes mis en œuvre concernent principalement l'apport de la sécurité dans la mise en œuvre des protocoles.

Sur un plan plus opérationnel la sécurité nécessite le déploiement de solutions reposant sur l'utilisation de multiples mécanismes de sécurité et dont la coordination est nécessaire. Il y a déjà à ce niveau une complexité qui apparait car il faut s'assurer de la compatibilité des mécanismes mis en œuvre. Par exemple dans le cadre de la mise en place d'un VPN, on peut toujours s'accorder sur le choix d'une solution aux deux extrémités. Mais il faut que les moyens de chiffrement soient rigoureusement les mêmes d'un côté et de l'autre, que les protocoles de tunnelisation retenus soient identiques.

Une complexité complémentaire vient du fait que les politiques de sécurité qui sont établies sont sujettes à des variations, voire à des changements. Tout changement de politique peut impacter des équipements et leurs configurations. Une situation qui était stable peut ne plus l'être suite à une reconfiguration.

Dans les infrastructures réseau, la construction d'une architecture sécurisée repose sur une coopération entre plusieurs équipements où plusieurs mécanismes de sécurité sont présents. Il n'existe pas de mécanisme de sécurité unique qui permette de protéger intégralement un réseau, une infrastructure ou un service. Bien au contraire, plusieurs briques de sécurité sont nécessaires. Une autre problématique vient donc du nombre important des mécanismes sous-jacents et des services de sécurité implémentés dans les équipements.

La modélisation que nous proposons dans cette thèse est une contribution pour traiter de ces différentes formes de complexité. Tenter d'assurer à un usager (totalement étranger au monde du réseau) que l'exigence de disponibilité concernant son serveur accessible derrière un firewall, un IDS, et la mise en œuvre d'un tunnel sécurisé peut conduire au recours à des spécialistes. Positionner sur une échelle de valeurs le respect des exigences conduit à l'audit. Tout audit nécessite d'une part une compétence mais également un outillage. Dans notre travail nous avons eu pour souci de démontrer que la modélisation par flux d'information peut conduire à la détection d'erreur de configurations ou de reconfiguration et en conséquence porter un avis critique et argumenté sur la compatibilité ou l'incompatibilité qui peut affecter un enchainement de mécanismes de sécurité dans un environnement réseau. Nous avons dans ce contexte approché cette problématique en essayant de nous abstraire des technologies existantes, pour proposer un système générique.

Chapitre 3. Les méthodes de détection de conflits entre configurations de sécurité

L'inconsistance de mécanismes de sécurité indique que deux règles/configurations de sécurité sont contradictoires ou en conflit. Une inconsistance peut apparaître lorsque deux règles d'un même mécanisme sur un même équipement sont incompatibles. Par exemple, un firewall possède deux règles de filtrage telles que l'une accepte tous les flux de données avec comme adresse IP source 10.0.0.0/8 et l'autre bloque certains flux de données avec pour adresse IP source 10.20.30.40. Ce type d'inconsistance est appelé inconsistance atomique par [Laborde et al. 2004].

La manière la plus simple de résoudre de tels conflits consiste à mettre œuvre un algorithme particulier qui amènera à une décision unique de la part de l'équipement de sécurité configuré. [Samarati et al. 2001] ont listé différents algorithmes possibles :

- Denials-take-precedence : les autorisations négatives prennent le pas sur les autorisations positives, Most-specific-takes-precedence : l'autorisation la plus spécifique est appliquée
- Priority level : un niveau de priorité est associé à chaque règle. Lorsque plusieurs règles sont possibles, celle ayant le niveau de priorité le plus élevé est choisie
- Positional : La première règle trouvée est appliquée
- Time-dependent : la dernière règle écrite est appliquée
- Etc.

Si ces algorithmes résolvent les conflits et donc garantissent que l'équipement n'aura qu'une unique règle à appliquer, ils ne permettent pas l'analyse et la détection a priori des inconsistances. Par ailleurs, ils ne garantissent pas la cohérence des règles qui seront appliquées car de plus, ce type de solution est limité à la résolution de conflit au sein d'un unique équipement. Ils ne s'appliquent pas à ce que [Laborde et al. 2004] appellent l'inconsistance distribuée, c'est à dire l'incompatibilité de deux règles/configurations qui peuvent être de mécanismes différents sur des équipements différents. Par exemple, lorsqu'un tunnel IPsec entre deux routeurs IPsec est « coupé » par un firewall intermédiaire. Ce type de problème est beaucoup plus difficile à gérer que le premier car il implique la prise en compte des dépendances entre des équipements différents et/ou des mécanismes différents.

Nous présentons dans ce chapitre différent travaux de recherche traitant de la détection et l'analyse de l'inconsistance atomique et de l'inconsistance distribuée.

3.1 Analyse de la consistance des règles de firewalls

Le firewall étant un des équipements les plus employés dans le cadre de la sécurisation des réseaux, les premiers travaux qui ont cherché à détecter des conflits de sécurité ont naturellement traité ce type de mécanisme.

Al-Shaer et Hamed ont proposé la première taxonomie permettant de classifier les différentes anomalies entre règles de firewall [Al-Shaer et al. 2004]. Leur approche consiste à modéliser formellement des règles de filtrage pour ensuite les analyser. Une règle de firewall sur du trafic IP est définie ainsi : <*order*><*protocol*><*sip*><*sport*><*dip*><*dport*><*action*> où <*order*> représente le niveau de priorité, <protocol> est le protocole encapsulé dans par le datagramme IP, <*sip*> et <*dip*> les adresses IP source et destination, <*sport*> et <*dport*> les ports source et destination et <*action*> peut prendre les valeurs accepter ou bloquer. En se basant sur cette représentation, ils définissent des relations entre règles de filtrage telles que l'inclusion, l'équivalence, la disjonction complète et partielle, ou encore la corrélation. Ces relations sont ensuite utilisées pour définir des types d'anomalies. Des algorithmes y sont décrits permettant de détecter ces anomalies.

Tout d'abord, Al-Shaer et Hamed ont étudié les **anomalies intra-firewall**, c'est à dire les conflits entre règles d'un même firewall :

- Les anomalies intra-firewall de type **occultation** : une règle A est dite occultée par une règle B si 1) B est plus prioritaire que A, 2) leur actions sont différentes (accepter/bloquer) et 3) tous les datagrammes correspondant à la règle A correspondent aussi à B. Ainsi, la règle A n'est jamais activée. Cette anomalie est considérée comme critique car cela peut engendrer le fait qu'un trafic réseau normalement accepté soit bloqué ou inversement qu'il soit accepté alors qu'il devrait être bloqué.
- Les anomalies intra-firewall de type **corrélation** : deux règles A et B sont corrélées si 1) leurs actions sont différentes (accepter/bloquer) et 2) une partie des datagrammes correspondant à la règle A correspondent aussi à la règle B. Cette anomalie n'est considérée que comme un avertissement.
- Les anomalies intra-firewall de type **généralisation** : Une règle B est une généralisation d'une règle A si 1) A est plus prioritaire que B, 2) A et B ont des actions différentes et 3) tous les datagrammes correspondant à A correspondent à B. Cette anomalie est un avertissement pour vérifier que la règle A est une exception de la règle B.
- Les anomalies intra-firewall de type **redondance** : Une règle A est dite redondante s'il existe une règle B telle que 1) les actions de A et B sont identiques, et 2) tous les

datagrammes correspondant à A correspondent à B. Ainsi, on peut supprimer une règle redondante sans modifier le comportement de filtrage du firewall. Ce type d'anomalie est considéré comme une erreur car il affecte les performances du firewall.

- Les anomalies intra-firewall de type **non-pertinence** : Une règle de filtrage est dite non-pertinente si elle ne peut s'appliquer à aucun trafic traversant le firewall. Tout comme les règles redondantes, les règles non-pertinentes dégradent les performances du firewall ainsi que la lisibilité de la configuration.

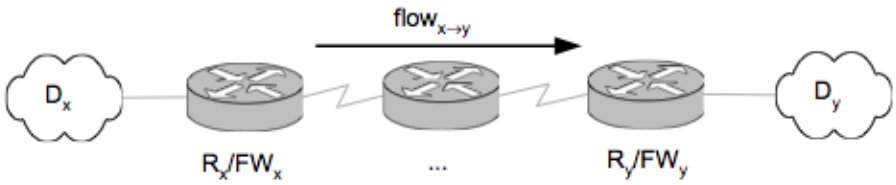

Figure 8. Firewalls en cascade [Al-Shaer et al. 2004]

Al-Shaer et Hamed ont aussi analysé les **anomalies inter-firewall**, c'est-à-dire entre des règles de différents firewalls. Pour décrire ces anomalies, ils considèrent la direction des flux de données par rapport aux adresses IP source et destination dans les règles de filtrage pour qualifier les firewalls de firewall amont et firewall aval. Par exemple dans la Figure 8, si les règles R_x et R_y portent sur des flux provenant du réseau D_x à destination du réseau D_y, alors le firewall amont est FW_x et le firewall aval est FW_y.

- Les anomalies inter-firewall de type **occultation** : Une telle anomalie se produit lorsqu'un firewall amont bloque un trafic réseau qui est accepté par un firewall aval.
- Les anomalies inter-firewall de type **non justification** : cette anomalie intervient lorsqu'un firewall amont accepte un trafic qui est par la suite bloqué par un firewall aval. Le firewall amont aurait dû le filtrer.
- Les anomalies inter-firewall de type **redondance** : cette anomalie est occasionnée lorsqu'un firewall bloque un trafic réseau qui est déjà bloqué par un firewall amont.
- Les anomalies inter-firewall de type **corrélation** : deux règles sur deux firewalls qui sont corrélées peuvent produire des ambiguïtés dans la politique de filtrage inter-firewall amenant des anomalies de type occultation et redondance.

Dans [Al-Shaer et al. 2005], les auteurs ont amélioré le travail précédent en permettant la détection mais aussi en proposant aux administrateurs des corrections d'anomalies. Ces algorithmes et des techniques ont été mis en œuvre dans un outil logiciel appelé le «Firewall Policy Advisor ».

Cuppens et al. [Cuppens et al. 2005 a] ont repris la taxonomie des anomalies de configuration de [Al-Shaer et al. 2004]. Ils ont généralisé la modélisation des règles de filtrage en ne limitant pas

uniquement la partie condition aux adresses IP, aux numéros de ports et au protocole encapsulé. Chaque règle de filtrage est définie comme suit:

$$R_i : \{condition_i\} \rightarrow décision_i$$

où *i* est la position relative de la règle à l'intérieur de l'ensemble de règles, $décision_i$ est soit accepter soit bloquer et $\{condition_i\}$ est un ensemble conjoint d'attributs $\{A_{1...p}\}$ tel que $\{condition_i\}$ égale $A_1 \wedge A_2 \wedge ... \wedge A_p$, « *p* » étant le nombre d'attributs de la condition. Ils ont aussi amélioré les algorithmes de détection en proposant des algorithmes de correction correspondant à des processus de transformation d'un ensemble initial de règles avec des anomalies politiques potentielles en un ensemble équivalent de règles sans erreur. Les règles qui en résultent sont complètement disjointes. Ce travail traite des anomalies intra-firewall de type occultation, redondance, non-pertinence et occultation. Le processus de correction a été automatisé dans [Cuppens et al. 2005 b] et mis en œuvre dans un prototype de logiciel pour démontrer la faisabilité de leur travail qui est basé sur un langage de script d'usage général [Castagnetto et al. 1999].

[Basile et al. 2004] ont proposé des algorithmes pour la découverte d'anomalies dans un firewall stateless, ainsi que lors de l'insertion (la mise à jour) de nouvelles règles. Ils ont développé la notion d'interprétation géométrique d'une politique de filtrage.

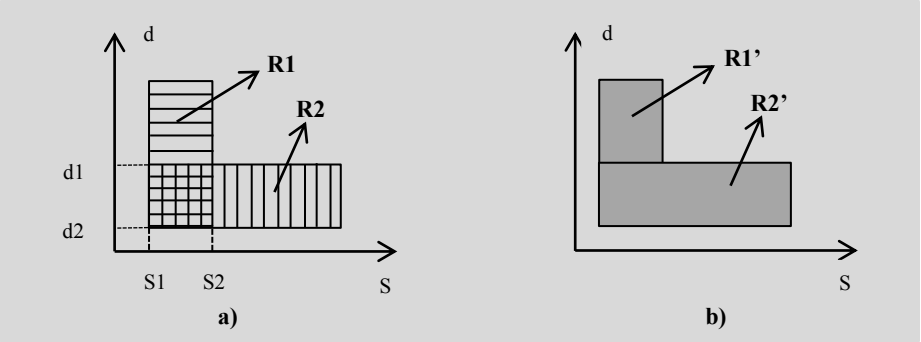

Figure 9. Réécriture en cas de redondance

Intuitivement, l'approche repose sur la projection d'une règle ayant « n » attributs sur un espace à n dimensions (Figure 9). En considérant deux règles de filtrage qui impliquent des adresses IP de la même sous-classe, l'abscisse constitue un intervalle d'adresses sources et l'ordonnée est un intervalle d'adresses destinations. Une règle apparaîtra dans cet espace comme un rectangle. Une anomalie existe si deux (ou plusieurs) rectangles sont superposés. Si on se limite à la redondance, l'algorithme de réécriture s'articule autour de la redéfinition des règles rectangles, qui n'auront aucune portion commune (totalement disjointes), conservant la même figure dans l'espace 2D. Dans l'exemple de la Figure 9, les règles R1 et R2 ont une « portion » commune (redondance partielle), sur l'espace d'adressage source [s1, s2], respectivement destination [d1, d2] (Figure 9.a). Les algorithmes de réécriture détectent la redondance et réalisent les règles R1', R2' qui sont disjointes dans l'espace 2D

(Figure 9.b). Par la suite, ils ont étendu ce travail [Basile et al. 2012] pour représenter les différentes anomalies de [Al-Shaer et al. 2004]. Ainsi, il est possible d'utiliser leur interprétation géométrique d'une politique de filtrage pour analyser les différentes anomalies entre règles stateless d'un firewall.

Les firewalls stateless ayant tendance à être remplacés par des firewalls stateful, des travaux plus récents ont tenté de détecter les anomalies en prenant en compte l'état d'une connexion.

Une première tentative pour modéliser des firewalls stateful a été effectuée dans [Gouda et al. 2005]. Dans ce modèle de firewall stateful, chaque firewall dispose d'un ensemble des variables appelées *état du firewall* utilisées pour stocker les datagrammes IP que le firewall a acceptés auparavant et dont il doit se souvenir pour de futures décisions.

Ils ont défini un paquet comme étant un d-uplet $(p_1,..., p_d)$ construit sur les champs $F_1, \ldots, F_d$ tel que chaque « p_i » est dans le domaine $D(F_i)$ du champ F_i et chaque $D(F_i)$ est un intervalle d'entiers positifs. Par exemple le domaine des adresses IP source est $[0,2^{32})$.

Chaque configuration de firewall se compose de deux sections : une section à état et une section sans état. Chaque section est une séquence de règles. Pour chaque datagramme IP, la section à état est utilisée pour gérer les états, c'est-à-dire vérifier si *l'état* comporte un datagramme précédent pouvant affecter la décision sur datagramme IP courant. Pour enregistrer le résultat de la vérification, ils ont supposé que chaque datagramme IP possèdait un champ supplémentaire appelé « tag ». Ainsi la section à état d'un firewall consiste en une séquence de règles à état comme suit :

$$P(F_1,..., F_d, F'_1,..., F'_d, tag') \rightarrow tag := x$$

Où $P(F_1, \cdots, F_d, F'_1, \cdots, F'_d, tag')$ est un prédicat sur $F_1, \cdots, F_d, F'_1, \cdots, F'_d, tag'$.

Un datagramme IP $(p_1, \ldots, p_d)$ correspond à une telle règle si et seulement si il existe un datagramme IP $(p'_1, \ldots, p'_d)$ avec un tag t' tel que $P(p_1, \ldots, p_d, p'_1, \ldots, p'_d, t')$ soit vrai. Lorsqu'il y a correspondance, le tag de ce datagramme $(p'_1, \ldots, p'_d)$ prend la valeur x.

La section *sans état* d'un firewall définit les décisions de filtrage par rapport aux états. Elle consiste également en une séquence de règles où chaque règle est appelée règle sans état. Une règle sans état est de la forme :

$$F_1 \in S_1 \wedge \cdots \wedge F_d \in S_d \wedge tag \in S_t \rightarrow < decision >$$

Où chaque S_i est un sous-ensemble non vide du domaine de F_i pour $0 \leq i \leq d$, S_t est un sous-ensemble non vide du domaine du champ « *tag* » et < *decision* > est « accept », ou « discard », ou « accept; insert »[1],. Par exemple, le champ port source appartient au domaine D(port source) = [0, 65 535]. Pour indiquer que le champ port source doit être inférieur à 1024 dans une règle de filtrage, on définit S_{port_source}=[0,1024] ⊂ D(port source).

[1] Accepter le datagramme et le rajouter à l'état du firewall.

Ces travaux ont fourni une première modélisation de firewall tout en considérant le suivi des connexions. Cependant, ils n'ont pas proposé de méthode pour analyser les anomalies de configuration.

En se basant sur ce travail, les mêmes auteurs [Gouda et al. 2007] ont remarqué que concevoir directement un firewall comme une séquence de règles présente trois types de problèmes majeurs :

1) le problème de consistance (consistency) : il est difficile d'ordonner les règles correctement

2) le problème de complétude (completeness) : il est difficile d'assurer un examen complet pour tous les types de trafic

3) le problème de compacité (compactness) : il est difficile de garder le nombre de règles faible parce que certaines règles peuvent être redondantes et certaines règles peuvent être combinées en une seule règle

Ils ont proposé une méthode consistant en deux étapes nommée : « structure firewall design » pour prendre en compte ces trois types des problèmes. Dans la première étape, ils analysent les propriétés de consistance et de complétude au travers d'un diagramme de décision d'un firewall FDD (firewall decision diagram). Un FDD *f* sur les champs $F_1, F_2, ..., F_d$ est un graphe orienté acyclique qui a cinq propriétés :

1) Il existe un unique nœud dans *f*, appelé racine, qui ne possède pas d'arc entrant

2) Chaque nœud *v* de *f* est étiqueté avec un champ noté *F(v)*. *F(v)* correspond à un champ pour un nœud non terminal ou {accept, deny} pour un nœud terminal

3) Chaque arc *e* de *f* sortant du nœud *v* est étiqueté avec un ensemble non vide d'entiers noté $I(e) \subseteq D(F(v))$ représentant les valeurs possibles de la condition de la règle sur le champ *F(v)*

4) Un chemin dans le FDD est appelé chemin de décision avec comme contrainte que pour tous nœuds v_1 et v_2 dans un chemin de décision $F(v_1) \neq F(v_2)$

5) L'ensemble des arcs sortants d'un nœud *v* de *f*, notés *E(v)*, doivent satisfaire les propriétés suivantes :

- Propriété de consistance : $\forall e, e' \in E(v), I(e) \cap I(e') = \varnothing$,
- Propriété de complétude : $\cup_{e \in E(v)}\{I(e)\}=D(F(v))$.

Dans la seconde étape, ils ont écrit un programme qui convertit un FDD en une séquence de règles de filtrage respectant la propriété de compacité. Cependant, cette méthode d'analyse n'est présentée que pour des firewalls stateless.

[Buttyan et al. 2009] ont réalisé un modèle formel pour la vérification d'un firewall à état. Ils ont modélisé un état comme étant un sous-ensemble particulier de l'ensemble des règles de filtrage d'un firewall correspondant aux règles activées. L'état du firewall est donc considéré comme un vecteur binaire représentant l'ensemble des règles dynamiques actives ou non dans la table de suivi de connexion comme illustré dans la Figure 10. Les auteurs se sont appuyés sur les travaux de FIREMAN [Yuan et al. 2006] qui est un outil pour tester des anomalies dans un environnement sans état. Une règle est dite sans état lorsqu'elle est représentée par < *P, action* > où « *P* » est le prédicat décrivant les critères que les paquets doivent satisfaire pour correspondre à la règle (c'est-à-dire le protocole, les adresses et ports sources et de destinations), et « *action* » est accepter ou bloquer. En se basant sur FIREMAN, les auteurs ont modifié la modélisation du travail précédent pour représenter les règles stateful de la forme suivante <*P, action, infoetat*> où *infoetat* est l'état de la connexion. Ils ont considéré les 4 types d'anomalies intra-firewall occultation, corrélation, généralisation et non-pertinence telles que définies par [Al-Shaer et Hamed, 2004]. Ils ont appliqué leur approche à iptables.

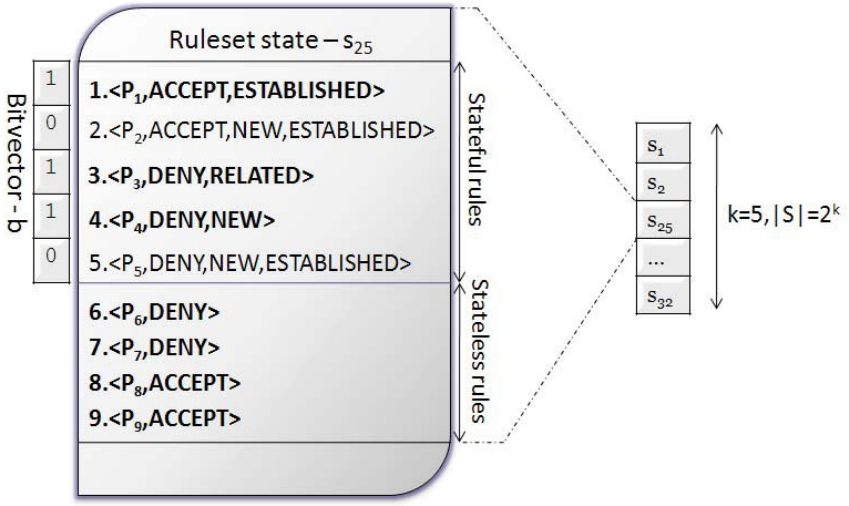

Figure 10. Encodage de l'état d'un firewall [Buttyan et al. 2009]

Cuppens et al. [Cuppens et al. 2012] ont présenté une alternative à la gestion des anomalies d'un firewall stateful. Les auteurs ont étendu l'algorithme proposé dans [Alfaro et al. 2008]. L'objectif de ce travail est de découvrir de nouvelles anomalies dans un environnement à état. Les anomalies découvertes sont classées ainsi :

- Les anomalies intra-état : Par exemple, l'établissement d'une connexion TCP se déroule en trois phases (SYN, SYN+ACK, ACK). Il est possible pour un firewall stateful de bloquer une de ces étapes
- Les anomalies inter-état : Ce type d'anomalie peut intervenir avec des protocoles comme FTP qui utilisent plusieurs connexions (une connexion de contrôle et une

connexion de données pour chaque transfert de données). Ici, l'état de la connexion de transfert de données dépend de la connexion de contrôle.

La communication entre deux nœuds avec un protocole de la couche transport est constitué de (1) la phase de mise en place « A_1 » et (2) la phase de terminaison « A_2 ». Ils ont caractérisé ces deux automates par :

- $\sum$ est l'alphabet de l'automate, qui contient une série de drapeaux « flags » d'état d'un protocole. Par exemple : pour TCP, $\sum$ = {SYN, SYN+ACK, ACK, FIN, FIN+ACK}
- Q contient les états des protocoles, comme les états CLOSED, LISTEN, and SYN SENT dans le cas de TCP
- δ définit la fonction de transition, comme: δ: Q x$\sum \rightarrow$ Q
- q_0 définit l'état initial. Pour TCP qui inclut LISTEN pour A_1 et ESTABLISHED pour A_2.
- q_f définit l'état final. Pour TCP, A_1 = ESTABLISHED; and A_2 = CLOSED.

Une configuration de filtrage est définie comme un ensemble de règles *R* de taille *n* contenant des règles R_i $(i \leq n)$. Chaque règle R_i spécifie une action (e.g., ACCEPT ou DENY) qui s'applique à un ensemble d'attributs de condition, comme : SourceAddr, DestAddr, SrcPort, DstPort, Protocol, Flag, State.

Ces différentes approches permettent de détecter voire corriger des anomalies de configuration. Cependant, elles se limitent soit à des firewalls stateless, soit à un unique firewall stateful. Plusieurs mécanismes de sécurité concourent à la protection d'un réseau, il est donc nécessaire de considérer ces différents équipements dans l'analyse de la mise en œuvre d'une politique de sécurité.

3.2 Analyse de la consistance des règles de plusieurs technologies de sécurité.

Les travaux de Fu et al. [Fu et al. 2001] exposent la nécessité d'un système de gestion pour assurer un service de sécurité de bout-en-bout en considérant deux mécanismes : le filtrage et les VPN IPsec[2]. Ils introduisent deux niveaux pour spécifier une politique de VPN IPSec :

- un niveau « haut » (high level policy) qui représente et analyse les objectifs de la politique IPsec
- un niveau « bas » (low level policy) qui correspond à l'implémentation de la politique dans les composants de sécurité. Ils ont développé des mécanismes pour détecter et

[2] Pour plus de détails sur IPsec se référer au chapitre 5.

résoudre les conflits entre les stratégies IPSec pour assurer des communications sécurisées de bout en bout

Un processus de transformation entre les niveaux doit être défini tout en gardant la conformité entre ceux-ci. Un même niveau « haut » peut se traduire en diverses configurations concrètes au niveau réseau, en se basant sur certaines hypothèses (par exemple, de confiance : une passerelle a le droit d'analyser le trafic, en le déchiffrant, car on lui fait confiance).

Ils ont présenté deux problèmes qui peuvent surgir lors de la mise en œuvre d'une architecture VPN IPsec. Le premier problème porte sur les conflits entre un firewall qui bloque un tunnel IPsec. Ce conflit peut être la conséquence de 1) soit une configuration incorrecte du firewall ou 2) soit le firewall, en analysant le trafic, bloque le tunnel car les données sont chiffrées.

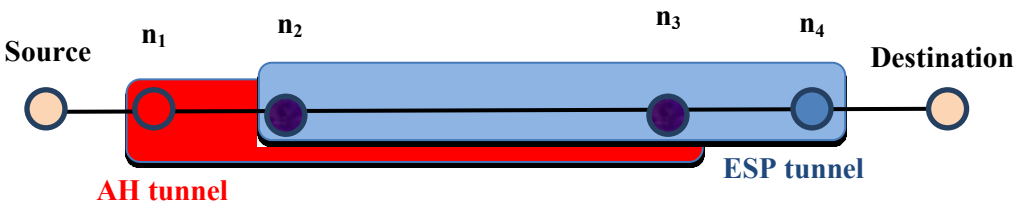

Figure 11. Chevauchement de tunnel IPsec

Le second problème est le chevauchement de tunnels IPsec. La Figure 11 présente un exemple où le trafic source-destination est encapsulé en mode tunnel deux fois : par le nœud n_1 (IPsec AH en mode tunnel entre les nœuds n_1 et n_3) et par le nœud n_2 (IPsec ESP en mode tunnel entre les nœuds n_2 et n_4). En raison de la deuxième encapsulation, le trafic est envoyé à n_4. Ce routeur désencapsule le protocole IPsec ESP ce qui donne l'entête AH avec l'adresse de destination n_3. Par conséquent, le trafic remonte à partir de n_4 jusqu'à n_3 où la deuxième décapsulation a lieu. Enfin l'adresse de destination est révélée et le trafic passe alors une seconde fois dans la section n_3-n_4 pour atteindre la destination. Seulement ce trafic n'est plus protégé par ESP (objectif du second tunnel) : la confidentialité n'est pas préservée. Une telle anomalie est le résultat du déploiement de tunnels IPsec qui se chevauchent.

Les spécifications de niveau « haut » comprennent quatre types d'exigences :

1) les exigences de contrôle d'accès qui limitent l'accès au trafic de confiance. Les auteurs désignent le trafic par un *flow-id* qui est un 6-uplet *(src-addr, dst-addr, src-port, dst-port, proto, [user id])*. Ainsi, ce type d'exigence est représenté par des règles : *flow-id* → *{accept, deny}*.

2) Les exigences de couverture de la sécurité correspondent à l'application de fonctionnalités de sécurité (e.g., authentification, chiffrement, etc.) pour protéger un flux de données. Il est aussi possible de spécifier des nœuds de confiance qui pourront accéder au trafic pour l'analyser (des firewalls, des IDS, etc.). Ce type d'exigence est

exprimé comme suit : *flow-id → enforce(fonctions de sécurité, niveau de protection, réseau source, réseau destination, nœuds de confiance).*

3) Les exigences d'accès au contenu décrivent des contraintes pour permettre aux nœuds de confiance d'accéder au trafic réseau. Ces exigences sont définies ainsi : *flow-id, fonctions de sécurité, nœuds de confiance → {accept, deny}*.

4) Finalement, les exigences sur les associations de sécurité traitent des contraintes sur les tunnels IPsec : flow-id, nœud$_1$ SA, nœud$_2$ SA → {accept, deny}.

Au niveau « bas », les politiques portent sur les champs src-addr, dst-addr, src-port, dst-port, proto, ah-hdr (en-tête AH), esp-hdr (en-tête ESP). Quant aux actions, elles peuvent être de trois types : deny (rejeter les paquets qui ne vérifient pas condition), allow, ou ip-sec action (sec-prot, algorithm, mode, from/to) où sec-prot indique le type de protocole AH ou ESP, algorithm définit les algorithmes pour la négociation ISAKMP, mode peut être transport ou tunnel, from/to sont les deux nœuds qui créent l'association de sécurité SA.

Une politique de sécurité est « correcte » si le niveau « bas » est en parfaite conformité avec le niveau « haut ». Elle présente des anomalies si le trafic n'est pas traité comme spécifié au niveau « haut » pour tous les nœuds où l'on déploie la politique VPN.

Al-Shaer et al. [Al-Shaer et al. 2006] ont critiqué l'approche de [Fu et al. 2001] car elle ne permet de découvrir que les chevauchements de tunnels IPSec. Tout comme dans leur travail sur les anomalies de configuration de firewall [Al-shaer et al. 2004] (cf section 3.1), les auteurs ont produit une classification des anomalies pouvant affecter les configurations IPSec (Figure 12). La partie « access list conflicts » regroupe les anomalies liées à la partie sélection/conditions des règles. Cette partie comprend la condition des règles de filtrage ainsi les règles IPsec indiquant que le trafic doit être protégé. Les anomalies répertoriées dans cette partie correspondent aux anomalies dans [Al-shaer et al. 2004]. La partie « map-list conflicts » liste des anomalies liées aux transformations de sécurité à appliquer sur un trafic donné. Par exemple, quelle transformation cryptographique dans IPsec ? Les anomalies de type « nested-session conflicts » comprennent les problèmes liés à l'ordre d'application des transformations de sécurité. Les chevauchements de tunnels IPsec appartiennent à cette classe d'anomalies. Les anomalies de type « multi-transform conflicts » apparaissent lorsque deux règles de transformation sont appliquées au même flux et que la protection offerte par la deuxième transformation appliquée est plus faible que la première.

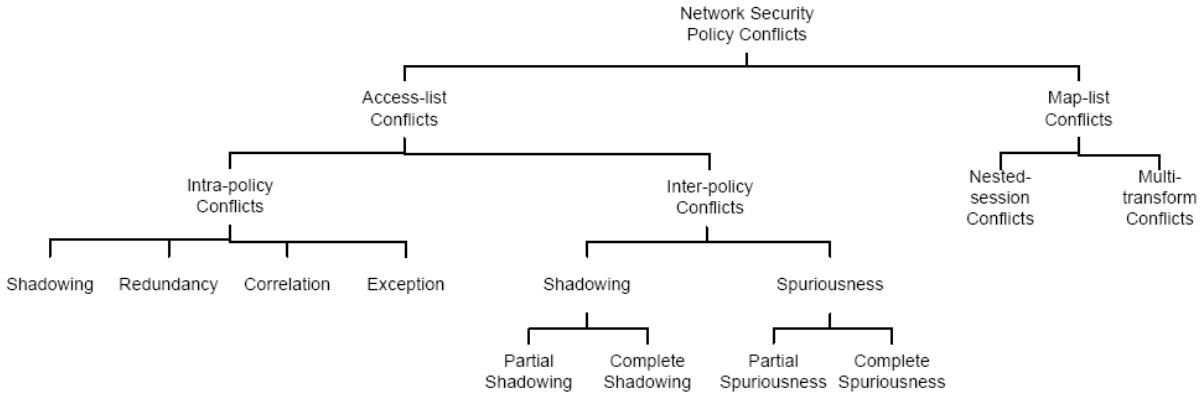

Figure 12. Diagramme de Classification de conflits de politique de sécurité réseau

Dans un article plus récent [Al-Shaer et al. 2009], les auteurs ont proposé une nouvelle approche dans le même contexte nommé « ConfigChecker| » qui permet de modéliser le réseau comme une machine géante à états finis où chaque état du réseau est déterminé par les datagrammes IP dans le réseau grâce à la fonction caractéristique $\sigma : (IPs \times ports \times IPd \times portd \times location) \rightarrow \{true\ or\ false\}$. Ils ont modélisé le comportement de chaque dispositif de contrôle d'accès en utilisant des diagrammes de décision binaires (BDD). Cette abstraction de réseau permet l'utilisation de techniques de model hecking contrôle pour vérifier les propriétés symboliques d'accessibilité et de sécurité écrites en CTL – Computational Tree Logic.

Ils ont étendu ce modèle pour intégrer certaines informations de charge utile qui résulte de l'encapsulation effectuée par IPsec. Pour ce faire, ils ont dû ajouter des variables supplémentaires pour résoudre le problème de la modélisation d'encapsulation IPsec (tunnel/transport, AH/ESP, etc.). L'encapsulation est gérée en cela en ajoutant des copies d'entêtes IP (IP source, port source, IP de destination et port de destination). Chaque copie comporte un bit de validité permettant de montrer que le datagramme a été encapsulé.

Une approche formelle au niveau des flux de données a été proposée par Guttman et Herzog [Guttman et Herzog 2005]. Le but de cette approche est de déterminer si une configuration du réseau incluant des firewalls et des passerelles IPsec vérifie bien les objectifs de sécurité. Tout d'abord, les auteurs modélisent un réseau en zones interconnectées par des nœuds qui mettent en œuvre les fonctionnalités de filtrage et IPSec. Ils définissent aussi un modèle abstrait pour les datagrammes IP par un triplet $<l,k,\theta>$ où l représente la localisation actuelle du datagramme, k l'état courant du traitement IPsec du datagramme et θ la suite d'entêtes IP représentant l'historique des actions effectuées sur le datagramme. Un entête IP est aussi vu de manière abstraite par un triplet $<s,d,p>$ où s et d sont respectivement la source et la destination et p le protocole encapsulé dans IP (protocole applicatif ou authentification (AH) ou confidentialité(ESP)). Une politique est définie comme étant une contrainte sur les trajectoires des datagrammes IP. Une politique de confidentialité est alors exprimée sous la forme suivante : si un datagramme provient de la zone A et par la suite atteint la

zone B, alors son θ doit contenir un entête fournissant la confidentialité. De même, les politiques d'authentification et de filtrage sont exprimées sous la forme de contraintes appliquées aux trajectoires des datagrammes IP entre deux zones.

Uribe et al. [Uribe et al. 2007] ont développé un outil de spécification et de vérification pour les réseaux qui comportent plusieurs firewalls et plusieurs IDS. Ce travail aide l'administrateur au placement de sondes IDS dans un réseau incluant des firewalls. Le langage de description formelle du réseau est basé sur le travail de [Guttman et al 2003] de part son orientation datagramme IP. Cependant la notion de datagramme est ici étendue avec la notion de « langage de paquet » noté *L*, qui correspond à un ensemble de datagrammes IP de la forme : *packet with [from_ip :F, to_ip :T, service :S, payload :P,...]* où chaque champ est associé avec un domaine fini. Le réseau est modélisé de la même manière de [Guttman et al. 2005], c'est à dire des zones interconnectées par des nœuds. Une trajectoire est ici spécifiée par un couple $<L,P>$ pour un langage de paquet *L* et un chemin $P=<e_1, \ldots, e_2>$ correspondant à une séquence d'arcs. Une fonction appelée packet(P) fournit l'ensemble des datagrammes IP pouvant emprunter ce chemin, $packets(P) \stackrel{def}{=} \bigcap_{e \in P} filter(e)$. Une trajectoire $<L,P>$ est qualifiée de faisable si $L \subseteq packets(P)$. En analysant les trajectoires des datagrammes, les auteurs peuvent déterminer les emplacements des sondes IDS. Ils ont proposé aussi une extension pour prendre en compte les mécanismes qui modifient les datagrammes. En effet, un IDS ne pourra pas analyser un flux chiffré. Pour cela, ils ont remplacé le couple $<L,P>$ pour représenter les trajectoires par une séquence sous la forme : $<<L_1,n_1>,<L_2,n_2>,\ldots,<L_k,n_k>>$ où chaque L_i décrit les datagrammes traversant le nœud n_i. Le fait qu'un datagramme soit chiffré est représenté par un champ particulier dans le langage de paquet.

Laborde et al. [Laborde et al. 2007] a proposé une approche de raffinement de politique en se basant sur le formalisme des réseaux de Petri colorés [CPN tools]. Le processus de raffinement est découpé en trois phases (Figure 13). L'expression des objectifs de sécurité réseaux qui sont déterminés à partir de politique de contrôle d'accès RBAC [Sandhu et al 1996] et [Ferraiolo et al. 2001]. La définition d'une tactique de sécurité qui est une représentation abstraite d'une solution pour mettre en œuvre les objectifs de sécurité. Une tactique de sécurité est indépendante de la technologie. La validation d'une tactique consiste à analyser sa consistance (pas de règles contradictoires) et son exactitude (représente-t-elle l'objectif de sécurité ?). Enfin, les configurations de sécurité qui représentent la mise en œuvre d'une tactique dans un environnement technologique cible. Le processus valide si une tactique peut effectivement être mise en œuvre dans un environnement donné.

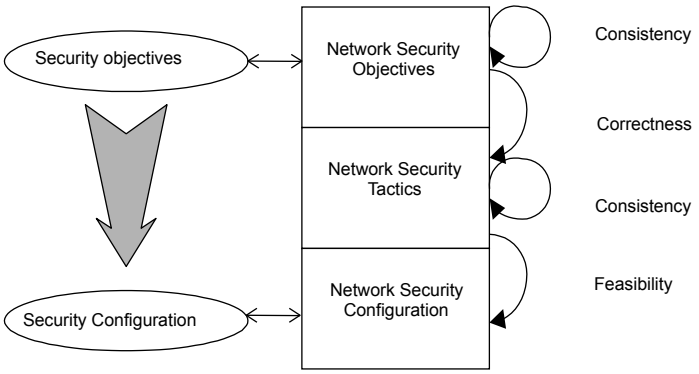

Figure 13. Processus de raffinement de Laborde

L'approche de modélisation consiste à déterminer les fonctionnalités de base dans la sécurité des réseaux qui lorsqu'elles sont combinées permettent de spécifier n'importe quel type d'équipement. Un flux de données peut être produit/consommé, propagé, transformé ou filtré indépendamment des technologies utilisées. Ce choix permet de limiter le nombre de cas à prendre en compte et donc de simplifier la spécification des tactiques de sécurité. Par conséquent, un équipement en vu non pas comme une entité propre mais par les traitements qu'il est capable d'appliquer sur les flux de données au moyen de quatre fonctionnalités de base (Figure 14) :

- Les fonctionnalités qui *produisent* ou *consomment* les flux de données comme les systèmes terminaux. Ces fonctionnalités sont appelées des **terminaisons de flux** (EF). Elles définissent le périmètre d'une structure de communication ; cela peut être un équipement terminal (par exemple un PC ou un équipement serveur), comme un processus applicatif (par exemple une application cliente ou serveur). Elles sont connectées aux entités sujets/objets du modèle RBAC. Une EF qui est associée à une entité active du modèle de niveau application (les utilisateurs dans le modèle RBAC), est appelée terminaison de flux active (AEF – Active End-Flow). Une EF qui est associée à une entité passive du modèle de niveau applicatif (les objets dans le modèle RBAC) est appelée terminaison de flux passive (PEF – Passive End-Flow). Les flux produits dépendent des permissions assignées aux rôles des sujets. Un ensemble de rôles est associé à chaque EF. Les rôles d'une EF proviennent des rôles de l'entité de niveau application qui est connectée. Un rôle associé à une EF représente ici simplement une abstraction des flux de données que l'EF peut produire et donc implicitement les objectifs de sécurité réseaux. Un flux est défini par *(efs, efd,rôle, liste_transf)* où *efs* est l'EF émettrice, *efd* est destinataire, *rôle* est le rôle caractérisant le flux de données, et *liste_transf* est la liste des transformations appliquées au flux de données.
- Les fonctionnalités qui *propagent* les flux de données comme les supports de communication. Ces fonctionnalités sont appelées des **fonctionnalités canal**. Les fonctionnalités canal représentent les environnements de propagation des flux de données

qui peuvent être physiques (bus de communication d'un ordinateur, câble, air pour le WIFI) ou une abstraction pour les systèmes non connus (Internet)

- Les fonctionnalités qui *transforment* les flux de données où l'on retrouve les protocoles de sécurité. Ces fonctionnalités sont appelées des **transformations**. Les fonctionnalités de transformation représentent la capacité de modifier un flux de données. Cela peut être un protocole de chiffrement comme IPsec où une fonctionnalité transforme un flux de données en y ajoutant un service de sécurité (par exemple la confidentialité) et une autre fonctionnalité enlève cette transformation. Cela peut être également le NAT en quel cas une seule fonctionnalité de transformation est impliquée et aucun service de sécurité n'est ajouté au flux de données. Les règles de transformations sont de la forme $\{ef_s\},\{ef_d\}$,rôle → groupe où $\{ef_s\}$ désigne l'ensemble des terminaisons de flux sources, $\{ef_d\}$ l'ensemble des terminaisons de flux destinations, *rôle* le rôle utilisé pour envoyer ce flux de données et *groupe* la transformation à effectuer
- Les fonctionnalités qui *filtrent* les flux de données comme les firewalls. Ces fonctionnalités sont appelées des **filtres**. Les fonctionnalités de filtre représentent la capacité de bloquer ou de laisser passer un flux de données. Elles représentent le service de contrôle d'accès. Il peut se situer à différents niveaux : les modules de contrôle d'accès des systèmes d'exploitation, les serveurs mandataires, les firewalls, ou encore le contrôle d'accès sur les commutateurs. Une règle d'une fonctionnalité de filtrage est un ensemble de 4-tuples de la forme *{efs},{efd},rôle,groupe* où $\{ef_s\}$ désigne l'ensemble des terminaisons de flux sources, $\{ef_d\}$ l'ensemble des terminaisons de flux destinations, *rôle* le rôle utilisé pour envoyer ce flux de données et *groupe* le dernier groupe de transformation appliqué.

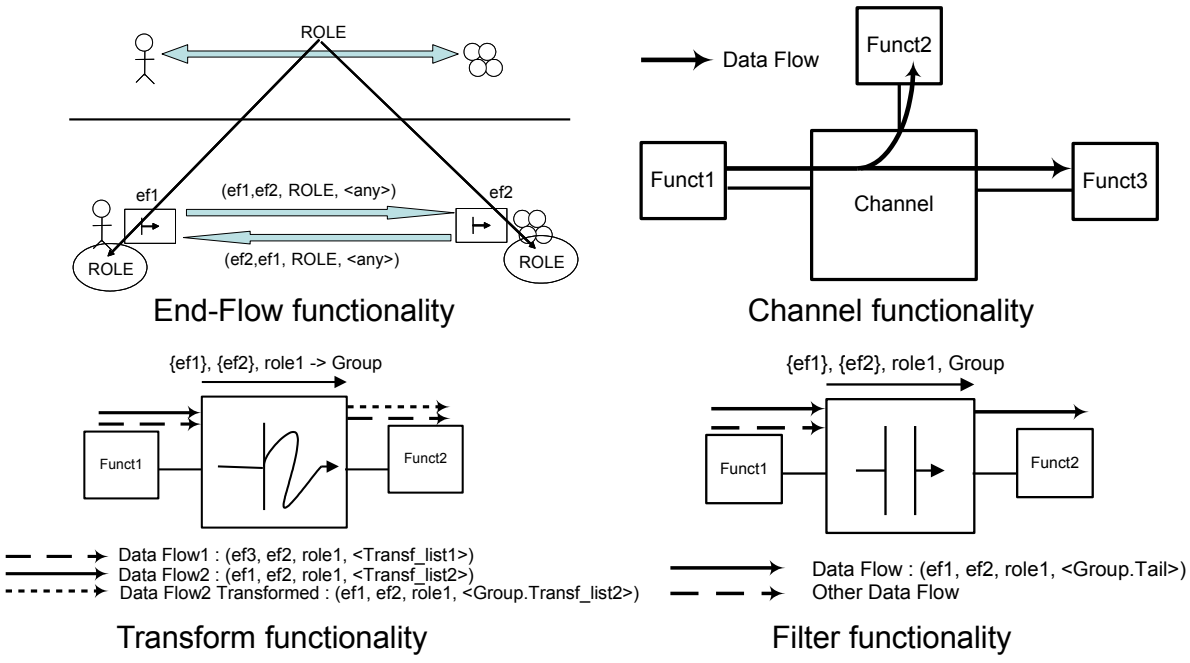

Figure 14. Fonctionnalités permettant de décrire les traitements sur un flux de données

Finalement, une tactique peut être validée automatiquement par rapport à différentes propriétés :

- Propriété de confidentialité : La propriété de confidentialité prévient de la divulgation non autorisée d'information. Dans cette modélisation, elle est exprimée par le fait que les terminaisons de flux actives ne doivent à aucun moment recevoir des flux de données lisibles avec des rôles qui ne leur ont pas été assignés
- Propriété d'accessibilité : Toutes les terminaisons de flux actives (resp. passives) doivent pouvoir consommer les flux de données pour tous les rôles qui leur sont assignés provenant de toutes les terminaisons de flux passives (resp. actives) assignées à ces mêmes rôles
- Propriété de cloisonnement : Un flux de données ne peut traverser un réseau que si ce dernier se trouve entre la source et la destination autorisées. Ainsi, une fonctionnalité de filtre ne doit laisser passer un flux de données que lorsqu'elle se trouve entre une source et une destination autorisées pour ce flux de données
- Règles de transformation et de filtrage non productives : Une règle de filtrage ou de transformation est dite non productive si elle n'est jamais utilisée par une fonctionnalité de filtrage ou de transformation

Preda [Preda et al. 2010] a aussi proposé un processus de raffinement. Partant des règles d'autorisation concrète OrBAC [Abou Al Kalam et al. 2003], il en déduit des configurations de firewall, de VPN IPsec et de système de détection d'intrusion en utilisant la méthode B classique [Atelier B]. Le modèle OrBAC propose de structurer une politique de contrôle d'accès en deux niveaux d'abstraction : 1) Le premier niveau reprend les ensembles des utilisateurs, des opérations et des objets que l'on retrouve classiquement dans les modèles de contrôle d'accès. 2) Un niveau plus abstrait, constitué d'un ensemble d'organisations, où sont traités des rôles, des activités et des vues qui représentent les abstractions respectives des utilisateurs, des opérations et des objets par rapport à une organisation donnée. Par exemple, un utilisateur est lié à un ensemble de rôles pour une organisation donnée. Il peut être assigné à d'autres rôles pour une autre organisation.

Preda considère que les règles OrBAC concrètes sont correctes par rapport aux règles OrBAC abstraites grâce au travail de Benaïssa et al. [Benaissa et al. 2006]. Le modèle de raffinement de Preda est basé sur huit machines B (Figure 15) :

- La machine POLICY représente la politique concrète OrBAC
- La machine NETWORK correspond à une représentation du réseau sous la forme d'un graphe où les nœuds sont soit les équipements ou des réseaux, et les arêtes sont les

connexions entre équipements. A Chaque arête est associée un poids qui modélise le coût d'utilisation du lien (similaire aux protocoles de routage dynamique)

- La machine UPDATE-PEP définit les équipements de sécurité présents sur le réseau (par exemple, firewall, passerelle VPN, IDS)
- Les machines PATH, WEIGHTED_FOREST, MIN_WEIGHT_LINK et PRIORITY_QUEUE sont utilisées pour calculer le chemin minimum entre les nœuds du réseau. Ce chemin déterminera les équipements de sécurité (éléments de la machine UPDATE_PEP) à configurer
- Finalement, la machine DEPLOYMENT représente le déploiement de la politique concrète OrBAC sur le réseau cible

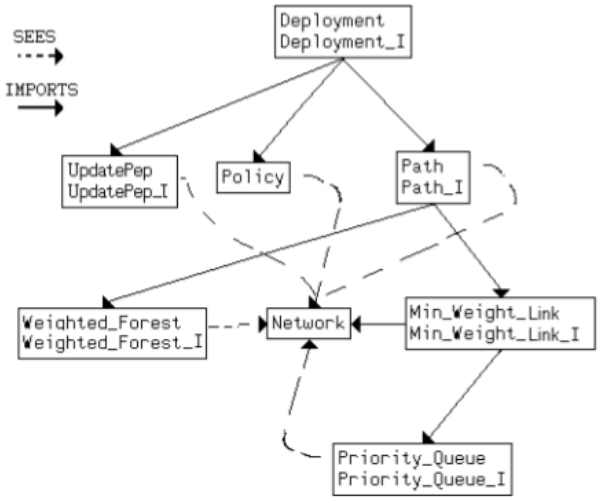

Figure 15. Graphe de dépendances des machines B

Le déploiement est analysé par rapport à un ensemble d'invariants :

- La complétude : si un chemin entre un sujet et un objet est correctement calculé (le chemin existe et les équipements de sécurité sur ce chemin possèdent les fonctionnalités requises) la règle de sécurité sera déployée
- L'accessibilité : Pour chaque permission, un sujet doit être capable d'accéder à un objet en respectant la politique
- Tout trafic doit être contrôlé par un firewall
- Propriété d'intégrité et de confidentialité : Si le trafic doit être protégé, il doit exister un tunnel IPsec

[Alfaro et al. 2008] ont proposé une démarche formelle pour détecter et corriger des anomalies entre firewalls stateless et sondes IDS. Leur approche reprend la taxonomie de [Al-Shaer et al. 2004] et la modélisation des règles de [Cuppens et al. 2005 a] [Cuppens et al. 2005 b]. Pour cela, leur

formalisme intègre dans les règles de configuration des composants (firewall ou IDS) une action appelée « alert ». Une étude sur la détection d'anomalies intra et inter composants basée sur ce formalisme y est développée. Dans MIRAGE [Alfaro et al. 2010], les travaux issus de [Preda et al. 2010] et [Alfaro et al. 2008] ont été implémentés pour garantir l'exactitude et la consistance des règles de configuration des politiques de sécurité réseau simples et distribués. Ils mettent en œuvre une analyse des configurations de composants de firewall, routeurs NIDS, et VPN pour détecter les anomalies de leur déploiement.

3.3 Conclusion

Différents travaux de recherche ont proposé de modéliser les différents conflits pouvant exister entre mécanismes de sécurité. Ils ont mis en relief la difficulté de mettre en œuvre une politique de sécurité sur les composants réseau. L'exploitation au quotidien de ces infrastructures montre des risques élevés d'erreurs et d'anomalies dans les configurations. Il est donc indispensable de disposer d'outils de vérification et de validation. Deux approches existent pour vérifier/valider des configurations de sécurité réseau mais elles rencontrent toujours des limites.

La première approche consiste à prendre comme élément de base le flux de données ou l'unité de données de protocole, e.g., le datagramme IP. L'avantage de cette approche est qu'elle est liée au modèle de flux de données et non à la technologie sous-jacente. Cependant, le niveau d'abstraction ne doit pas être trop éloigné de la réalité comme [Laborde et al. 2007]. De plus, le modèle de flux de données doit être extensible contrairement à [Guttman et al. 2005], [Uribe et al. 2007] ou [Al-Shaer et al. 2009].

La deuxième approche se focalise sur la modélisation des configurations des équipements ; la notion de flux de données étant secondaire. Les algorithmes développés sont alors très liés à la sémantique des éléments de configuration et en conséquence, les modèles sont liés à une ou plusieurs technologies précises et il est très difficile de les adapter à de nouvelles technologies [Fu et al. 2001], [Al-Shaer et al. 2004, 2005 et 2006], [Gouda et al. 2005 et 2007], [Alfaro et al. 2008 et 2010] et [Cuppens et al. 2012] etc.

Chapitre 4. Un modèle de mécanisme de sécurité générique basé sur les flux de données

4.1 Introduction

Notre objectif est de définir une modélisation des aspects de sécurité des réseaux pour permettre de comprendre le fonctionnement d'un réseau afin de pourvoir déceler des anomalies de configurations ayant un impact sur sa sécurité. Ces anomalies doivent pouvoir être décelées soit en amont de la phase de mise en œuvre des configurations (nouvelle configuration/mise à jour), soit en aval pour pouvoir remonter à la source d'un incident de sécurité lorsqu'il a été détecté. Laborde et al. [Laborde et al. 2004] ont listé différents éléments qu'une méthode formelle doit pouvoir prendre en compte :

1) *les différents éléments constituant un réseau*, i.e., les équipements terminaux, les équipements intermédiaires, les utilisateurs, les services applicatifs, etc. ;
2) *les différentes technologies* implémentées sur les éléments du réseau assurant des services de sécurité, par exemple les équipements de filtrages comme les firewalls, les systèmes de préventions d'intrusion, les serveurs mandataires ou encore les protocoles de sécurité de niveau données (ex. PGP), de niveau application (ex. SSH), de niveau transport (ex. SSL), de niveau réseau (ex. IPsec), de niveau liaison de données (ex. WEP) ;
3) *le plan d'interconnexion* des éléments du réseau, car la topologie d'un réseau a un impact important sur les effets des services de sécurité ;
4) *une abstraction adaptée* à différents besoins. Si dans certains cas, il est possible et même souhaitable de spécifier finement les éléments du réseau et leur topologie. Dans d'autres cas, ceci est impossible car il y a trop d'éléments à prendre en compte ou que ces éléments sont inconnus, par exemple le réseau Internet.

De plus, il est souhaitable que la spécification soit la plus simple possible afin d'éviter au maximum des erreurs de spécification. Par conséquent, tous ces « détails » comme la topologie du

réseau, les mécanismes de routage, de filtrage ou encore de transformation de flux de données doivent alors être pris en compte de manière générique.

L'étude des travaux sur la détection et la correction des anomalies de configuration de sécurité réseau a montré qu'il existe deux approches :

1) Les approches qui raisonnent sur les configurations. Ces méthodes offrent des outils d'analyse poussés et proches des technologies réelles. Cependant, elles sont limitées lorsque plusieurs technologies cohabitent, surtout lorsque ces technologies ont un impact en terme de sécurité à différents niveau de la pile TCP/IP. De plus, elles n'offrent pas la capacité de choisir son abstraction.
2) Les approches orientées flux de données permettent d'être indépendant des technologies. Cependant, si la modélisation des flux de données est trop spécifique, cette propriété n'est plus vraie. A contrario, si le modèle de flux de données est trop abstrait, il peut y avoir un écart trop important entre le niveau d'abstraction de la spécification et celui de la réalité.

Nous présentons dans ce chapitre une modélisation orientée flux de données. Chaque mécanisme de sécurité sera vu comme un traitement particulier sur les flux de données. Dans un premier temps, nous présentons notre modèle de flux de données. Ce modèle permet de représenter l'état d'un flux de données en considérant les différentes couches protocolaires. Dans un deuxième temps, nous décrirons les différents traitements possibles sur un flux de données. Notre idée est de lister les traitements de bases pouvant être effectués sur un flux de données pour ensuite représenter le traitement d'un mécanisme de sécurité par rapport à ces traitements de base. Enfin, nous décrirons une représentation générique d'un mécanisme de sécurité.

4.2 Notre inspiration pour la modélisation D'un flux de données

Afin de proposer un modèle de flux de données, nous évoquons quelques éléments fondamentaux sur la construction d'un flux de données. Dans cette section, nous effectuons quelques rappels sur les principaux modèles structurant les architectures réseaux (ISO, IEEE, TCP/IP) (Figure 16) ainsi qu'une analyse d'un flux de données au travers de l'outil WireShark [wireshark].

4.2.1 L'encapsulation de protocoles

Différents modèles et suite protocolaires ont été proposés pour structurer les systèmes afin qu'ils puissent communiquer au travers d'un réseau. Le modèle de référence ISO (Interconnexion de Systèmes Ouverts) proposé par l'organisation OSI prend ses origines de deux avancées des années 70 : 1) l'émergence de techniques structurées en couches dans la conception des réseaux, et 2) la reconnaissance du besoin d'architectures compatibles entre différents fabricants. Les différentes normes ISO ont donc proposé une architecture de systèmes ouverts (communicants) en 7 couches

permettant de décomposer ces systèmes complexes en parties plus petites et donc plus compréhensibles. Chaque partie est responsable d'un problème particulier. La communication entre couches adjacentes d'un même système ouvert s'effectue à travers la technique d'encapsulation. Cette technique consiste à inclure des paquets de données dans d'autres paquets de données. L'ISO a formalisé cette technique dans le cadre des réseaux en définissant qu'un paquet de données d'une couche N s'appelle (N)-PDU. Lorsque cette couche N désire utiliser les services de la couche N-1, ce paquet de données (N)-PDU et renommé (N-1)-SDU. Pour effectuer son service, la couche N-1 rajoute un paquet de données appelé (N-1)-PCI pour former un (N-1)-PDU en utilisant la technique d'encapsulation (Figure 17).

Même si le modèle ISO n'est pas mis en œuvre tel quel sur nos équipements. Les piles de protocoles IEEE et IETF respectent toutes ces notions de couche et d'encapsulation. Même les tendances actuelles consistant à faire passer tout protocole au-dessus du protocole HTTP (par exemple RPC over HTTP [Microsoft-1] [Msexchange], Java over HTTP [Microfocus], SOAP [Web Services 2008], ou encore les APIs REST [Microsoft-2]) gardent la notion d'encapsulation même si la notion de couche bien définie comme dans l'ISO est cassée.

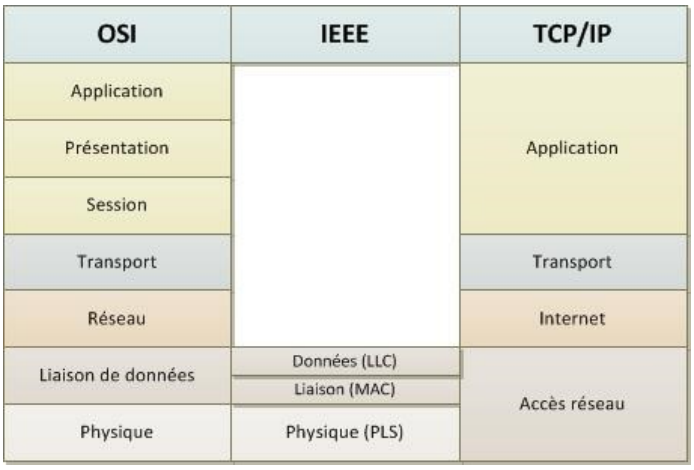

Figure 16. Modèle hiérarchique de base

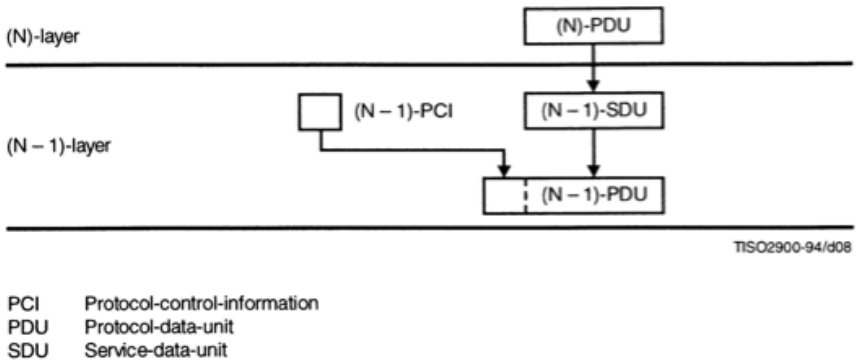

Figure 17. Transfert d'information entre couches ISO [ISO 7498-1]

4.2.2 Analyse d'un trafic réseau au travers de wireshark

Pour pouvoir analyser finement le trafic réseau et déboguer des problèmes réseau, il existe des logiciels de capture de trames qui sont des outils permettant de récupérer les paquets qui passent physiquement par une carte réseau (quelque soit la destination de ces paquets). L'interface de visualisation d'un trafic réseau au travers de l'outil Wireshark[3] comporte trois volets (Figure 18) :

- Le **volet 1** permet de recenser l'ensemble des trames capturées en affichant des informations basiques.
- Le **volet 3** présente les données réellement transmises sur le réseau au format hexadécimal et ASCII. Nous avons mis ici en évidence les champs du protocole Ethernet II avec 14 octets en début de trame correspondant (adresse MAC destination, adresse MAC source, identifiant du protocole encapsulé) et 4 octets pour le champ de contrôle à la fin.
- Le **volet 2** qui est encadré en rouge permet de visualiser l'encapsulation des protocoles de manière structurée. L'encapsulation de protocoles est représentée en colonne. Par exemple, un flux de données correspondant à une requête HTTP est exprimé sous la forme suivante <*bloc protocole Ethernet, bloc protocole IP, bloc protocole TCP, bloc protocole HTTP*>. Chaque élément protocole peut être déployé pour lister les champs de l'entête du protocole. L'emplacement physique des champs dans la trame n'est pas suivi. Par exemple, le bloc Ethernet comprend tous les champs. Le CRC qui normalement se place en fin de trame est ici recensé avec les autres champs. Ce volet 2 est donc une vue logique facilitant la compréhension des informations.

[3] **Wireshark** Interface graphique de capture et de décodage de trames. Propose les mêmes fonctions que « tcpdump » mais avec une interface graphique qui permet de visualiser directement les différents protocoles utilisés

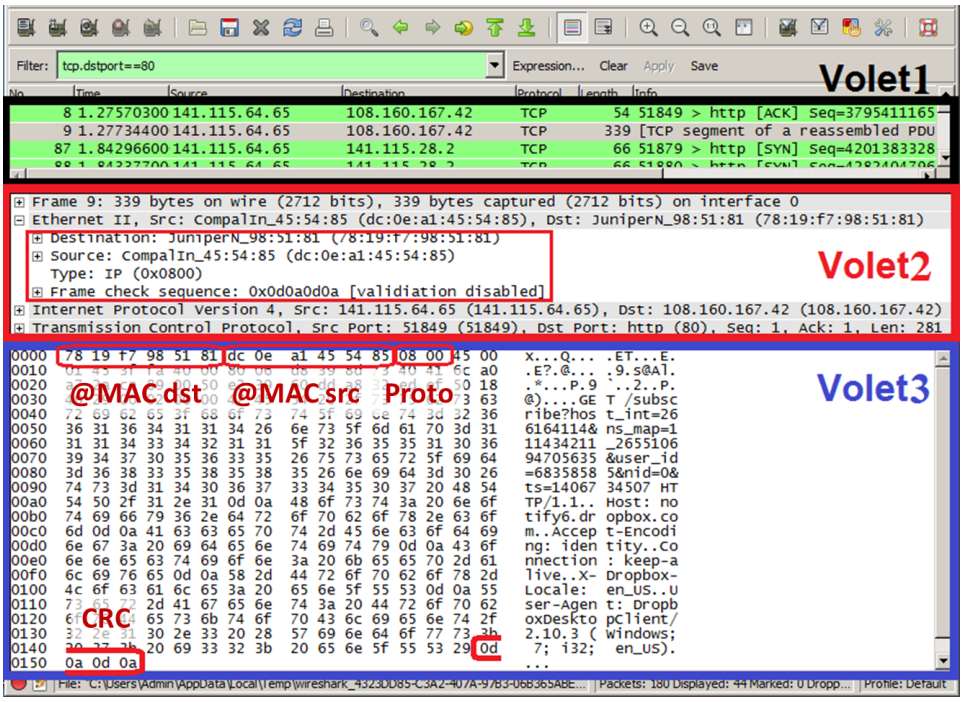

Figure 18. Capture d'image Wireshark

La Figure 19 présente une capture d'un trafic ICMP encapsulé dans un tunnel IPsec AH qui protège l'intégrité. La visualisation de wireshark montre bien l'ensemble de la pile de protocole avec les différents champs du protocole AH. Cependant, l'outil ne fournit aucune information sur les champs authentifiés ni sur les problèmes d'intégrité qu'il pourrait y avoir.

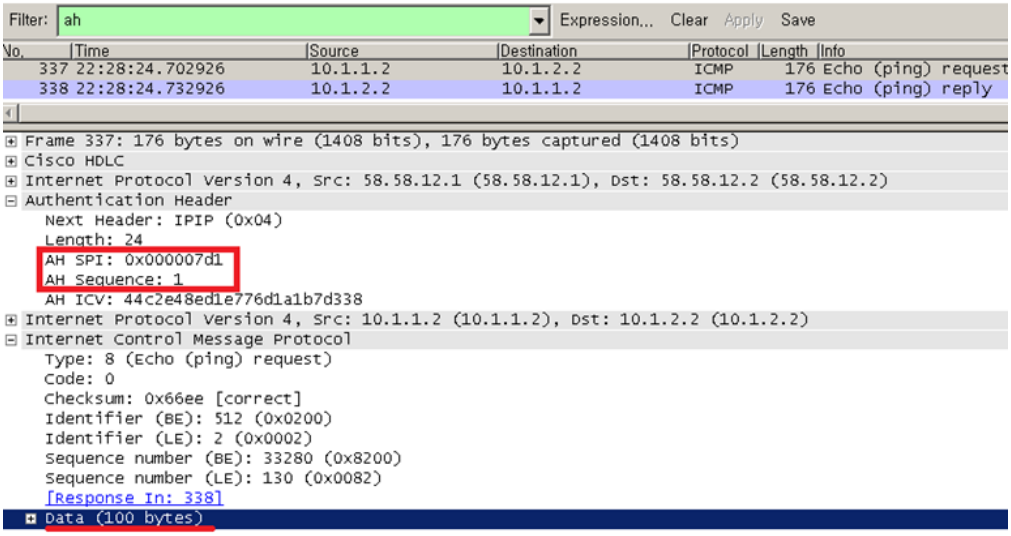

Figure 19. Exemple de flux authentifié par AH mode tunnel[4]

La Figure 20 est une capture d'un trafic protégé par le protocole IPsec ESP. Les données encapsulées dans le protocole IP ont été chiffrées. Toutefois, il est possible de pouvoir visualiser les

[4] source de la figure : http://sola99.tistory.com/153

données chiffrées en ajoutant les clés de chiffrements de l'association de sécurité à wireshark. Cependant, tout comme pour l'authentification, wireshark n'indique pas les champs chiffrés à l'utilisateur.

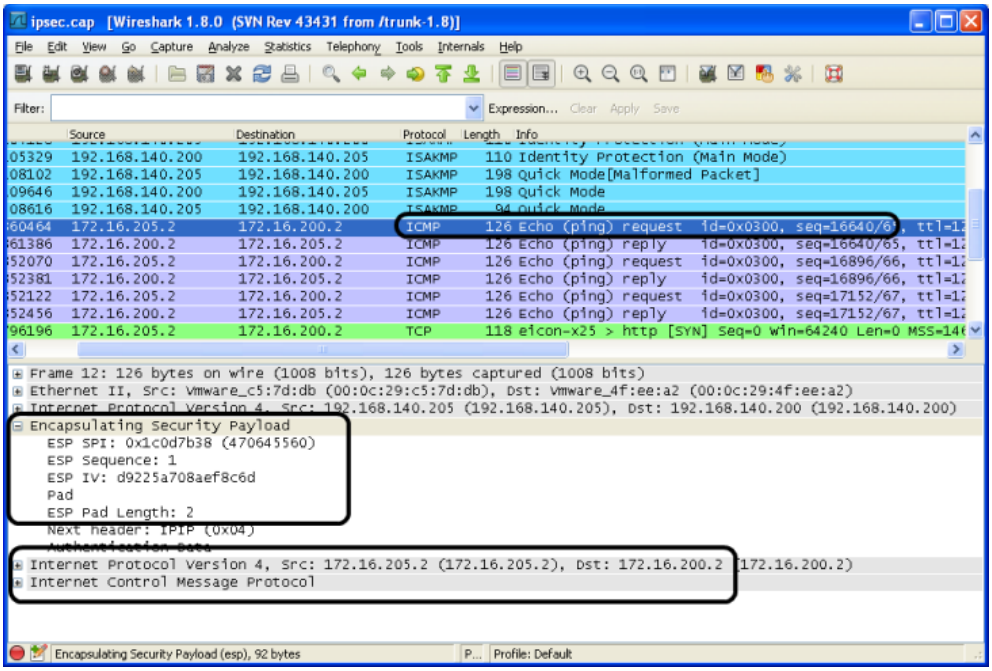

Figure 20.Exemple de flux chiffré par ESP[5]

L'outil wireshark est un des outils classiques pour l'analyse de trafic réseau en particulier lors de la phase de débogage. Il offre une vue structurée qui facilité la visualisation. La pile de protocoles encapsulés est représentée de manière logique par bloc de protocoles où tous les champs d'un même protocole sont regroupés dans le même bloc. Cependant, cet outil n'offre aucune information sur les traitements de sécurité qui ont été appliqués au trafic réseau. L'administrateur réseau doit alors posséder des connaissances pointues sur ces protocoles pour pouvoir effectuer une quelconque analyse.

4.2.3 Notre modélisation formelle d'un flux de données

Un flux de donnée est représenté comme étant une chaine d'encapsulations ou une suite d'éléments nommée communément « protocole ». Un flux de données envoyé sur le réseau sera modifié par les différents mécanismes intermédiaires. Certains de ces mécanismes appliqueront sur les flux de données des protections qui authentifieront et/ou chiffreront tout ou une partie du flux de données. Dans un objectif d'analyse, il nous faut garder l'historique des actions réelles effectuées sur ce flux contrairement à ce que peut faire « wireshark ».

Nous proposons un modèle de flux de données indépendant des protocoles sous-jacents. Nous désirons cette généricité afin de pouvoir anticiper les protocoles futurs. La difficulté consiste à trouver

[5] source de la figure : http://ask.wireshark.org/questions/12019/how-can-i-decrypt-ikev1-andor-esp-packets

le bon niveau d'abstraction permettant à la fois d'être indépendant des mécanismes de sécurité et de représenter de manière fidèle la réalité.

Tout d'abord, nous définissons nos éléments de base :

- $\mathcal{A}$ est l'ensemble des attributs possibles. Un attribut $a \in \mathcal{A}$ est un couple <*name, value*> tel que *name* est un champ quelconque que l'on peut trouver dans un protocole et *value* est son contenu,
- $\mathcal{P}$ est l'ensemble des protocoles, c'est à dire l'ensemble des blocs logiques protocole. Ainsi un protocole $p \in \mathcal{P}$ est vu comme un couple <*protoid, attributes*> tel que *protoid*=<*name, id*> contient le nom du protocole *name*, son identifiant unique *id* et *attributes* est l'ensemble des attributs de p et est défini sur l'ensemble des parties de $\mathcal{A}$, i. e. , $attributes \in \mathbb{P}(\mathcal{A})$,
- $\mathcal{E} = \mathcal{P}^{\mathbb{N}}$ est l'ensemble des séquences finies sur $\mathcal{P}$. Cet ensemble représente l'ensemble des chaînes d'encapsulation de protocoles, i.e., des séquences de blocs logiques protocoles. Pour des raisons de simplification de notation, nous utilisons $\text{succ}(p_i) = p_{i+1}$ et $\text{suite}(p_i) = < p_{i+1}, \dots, p_n >$ pour une séquence $p = < p_1, p_2, \dots, p_n >$ donnée,
- $\mathcal{S}$ est l'ensemble des algorithmes de sécurité traitant les chaînes d'encapsulation de protocoles (par exemple, DES, 3DES, HMAC-SHA1, etc).
- $\mathcal{L}$ ={*all*,*val*} représente le niveau de chiffrement d'un attribut. La valeur *all* indique qu'il est impossible de déterminer où se trouve l'attribut dans le flux. Par exemple, lors de l'utilisation ESP [RFC 4303], l'emplacement des champs des protocoles protégés par ESP ne peuvent être déterminés. Par contre, d'autres mécanismes comme XML-Encryption [XMLenc 2011] permettent de chiffrer soit tout un bloc XML soit uniquement un élément du bloc XML. Dans ce dernier cas, il est possible de déterminer l'emplacement de l'attribut contrairement au cas ESP, mais il est normalement impossible de connaître sa valeur sauf si la clé secrète est connue. Pour modéliser cette possibilité nous utilisons la valeur *val* qui indique que l'emplacement de l'attribut peut être déterminé mais pas sa valeur sauf si la clé est connue.

Nous avons ajouté à notre modèle deux ensembles AUTHN et CONF qui caractérisent l'historique des traitements liés à l'authentification et la confidentialité d'un flux de données.

Définition 1. ***Un flux de données***

A partir de ces définitions, nous présentons l'ensemble des flux de données comme étant :

$\mathcal{F} \subseteq \mathcal{E} \times \mathbf{AUTHN} \times \mathbf{CONF}$ tel que :

- $\mathcal{E}$ est l'ensemble des chaînes d'encapsulation,
- $\mathbf{AUTHN} \subseteq (\mathcal{A} \times \mathcal{P} \times \mathcal{A} \times \mathcal{P} \times \mathcal{S})$ représente les attributs du flux de données qui ont été authentifiés tel que $(a_1, p_1, a_2, p_2, s) \in$ AUTHN indique que l'attribut a_1 du protocole p_1 garantit l'intégrité de l'attribut a_2 du protocole p_2 via le algorithme de sécurité s,
- $\mathbf{CONF} \subseteq$ BAG$(\mathcal{A} \times \mathcal{P} \times \mathcal{S} \times \mathcal{L})$ représente les attributs du flux de données qui ont été chiffrés, tel que :
  - $(a, p, s, \text{all}) \in$ CONF indique que l'attribut a du protocole p est complètement chiffré par l'algorithme de sécurité s et
  - $(a, p, s, \text{val}) \in$ CONF indique que la valeur de l'attribut a du protocole p est chiffré par l'algorithme de sécurité s.

Nous avons choisi d'utiliser des multi-ensembles[6] pour CONF car un attribut peut être chiffré plusieurs fois par le même algorithme.

Exemple 4.1:

Soit un flux de données $f = (\text{e}, \text{AUTHN}, \text{CONF}) \in \mathcal{F} \mid \text{e} =< p_1, p_2 >,$ où :

$p_1 = ((p, 1), \{(\alpha, v1), (\beta, v2), (\gamma, v3)\})$ *et* $p_2 = ((p, 2), \{(\kappa, v4), (\mu, v5)\})$:

1) Si $\text{AUTHN} = \{\}$ et $\text{CONF} = \{\}$ Alors ce flux de données comprend deux protocoles encapsulés pour lequel aucun champ n'est protégé

2) Si $\text{AUTHN} = \{(\kappa, p_2, \gamma, p_1, s_1), (\kappa, p_2, \mu, p_1, s_1)\}$ et $\text{CONF} = \{(\mu, p_2, s_2, \text{all})\}$. Alors le champ κ dans le protocole p_2 authentifie les champs γ et μ dans le protocole p_1 via l'algorithme s_1 et le champ μ dans le protocole p_2 est totalement chiffré via l'algorithme s_2.

Un des avantages des ensembles AUTHN et CONF est la possibilité d'analyser et de conclure d'autres propriétés comme l'intégrité.

[6] Un multi-ensemble est un ensemble d'éléments où un élément peut apparaître plusieurs fois.

Définition 2. ***Un flux de données intègre***

Un flux de données intègre indique qu'aucun attribut authentifié n'a été modifié. Cela s'exprime dans notre modèle sous la forme suivante :

Soit le flux $\oint = (\mathrm{e}, \mathrm{AUTHN}, \mathrm{CONF}) \in \mathcal{F}$, $\oint$ est intègre si et seulement si :

$\forall(\mathrm{a}, \mathrm{a}') \in \mathcal{A}\ \mathrm{x}\ \mathcal{A}, \forall\ (p_1, p_2) \in \mathcal{P}\ \mathrm{x}\ \mathcal{P}, \forall\ \mathrm{s} \in \mathcal{S}, (\mathrm{a}, p_1, \mathrm{a}', p_2, \mathrm{s}) \in \mathrm{AUTHN} \Longrightarrow$
$\mathrm{a} \in \mathrm{attributes}(p_1)\ \wedge \mathrm{a}' \in\ \mathrm{attributes}(p_2)$

Nous ne fournissons pas de définition générique de la confidentialité d'un flux de données. La confidentialité fait référence à la non-divulgation d'informations sensibles. Les informations sensibles dans le contexte d'un flux de données sont un sous-ensemble des champs des protocoles qui doivent être confidentiels. Dans notre modèle de flux de données orienté attributs, les attributs du multi-ensemble CONF représentent l'information chiffrée dans le flux de données. Il faut noter que notre modèle représente fidèlement la réalité. Un flux de données sur un réseau ne peut pas être entièrement chiffré (si le champ de destination IP est chiffré, les routeurs ne seront pas capable de router le datagramme IP). Toutefois, des valeurs particulières d'attributs spécifiques peuvent être considérées comme confidentielles (e.g., si le fait de savoir que deux équipements communiquent sur Internet est confidentiel, il est important de cacher les valeurs de leurs adresses IP, par exemple dans un tunnel IPsec, où seules les adresses IP de deux passerelles IPsec seront révélées). Ainsi, l'ensemble des attributs devant être confidentiels dépend d'exigences de sécurité. Ainsi, le fait de capturer cette information, nous permet d'analyser de telles exigences de confidentialité.

Illustration d'une encapsulation:

En appliquant un exemple d'encapsulation réel (Figure 21) sur la théorie précédente $\oint = (\mathrm{e}, \mathrm{AUTHN}, \mathrm{CONF}) \in \mathcal{F} \mid \mathrm{e} =< p_1, p_2, p_3, p_4 >$, où :

$p_1 = \big((802.3, 1), \{\ldots\}\big), p_2 = \big((\mathrm{IP}, 2), \{\ldots, (\mathrm{ips}, 1.1.1.1), (\mathrm{ipd}, 1.0.0.0), \ldots\}\big),$

$p_3 = \big((\mathrm{TCP}, 1), \{\ldots, (\mathrm{ports}, 8080), (\mathrm{portd}, 88), \ldots\}\big)\mathrm{et}\ p_4 = ((\mathrm{HTTP}, 1), \{\ldots\})$:

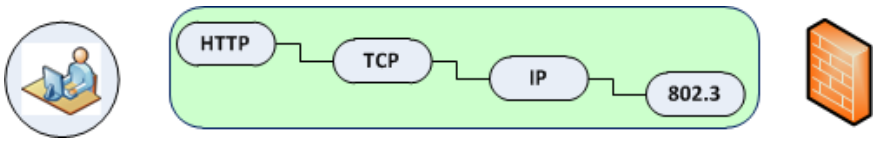

Figure 21. Encapsulation tcp/ip

Illustration de l'utilisation des ensembles AUTHN et CONF

L'historique des actions d'authentification et de confidentialité effectués sur un flux de données est maintenu avec les ensembles AUTHN et CONF. Il devient alors possible d'analyser le flux et ses propriétés. En plus, il nous permet de connaître à un instant « T » la valeur de n'importe quel attribut et son niveau de protection. Si l'on considère le protocole AH [RFC 4302], son attribut nommé *AD* (Authentication Data), garantit l'intégrité de la plupart des attributs du protocole IP et ceux qui sont supérieurs.

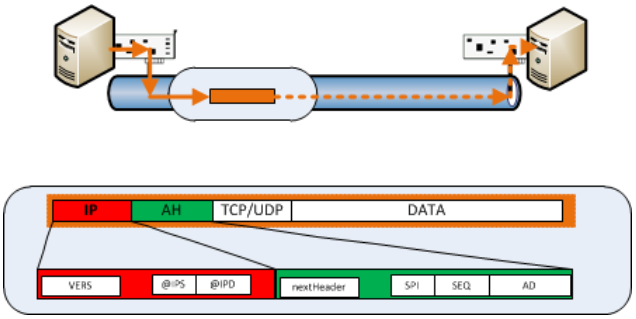

Figure 22. Datagramme IP protégé par AH en mode transport

Dans la Figure 22, nous présentons un flux de données encapsulé par le protocole AH. L'ensemble AUTHN sera rempli par l'ensemble des attributs qui sont authentifiés par l'attribut AD du protocole AH. La Figure 23 est l'application de l'exemple de la Figure 22 et présente les attributs. On remarque que l'attribut « AD » du protocole AH garantit l'intégrité des attributs « VERS, @IPS et IPD » du protocole IP en utilisant l'algorithme de sécurité hmac_ALGO, et ainsi de suite.

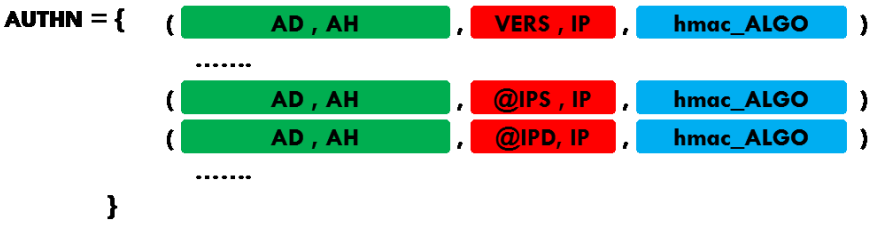

Figure 23. L'ensemble AUTHN

Si le traitement consiste en l'utilisation du protocole ESP, cela revient à ajouter de nouveaux protocoles (Figure 24) dans la chaîne d'encapsulation (ESP Header, ESP Trail et ESP Authentication) ainsi que de nouvelles instances dans la relation AUTHN et de nouveaux attributs dans le multi-ensemble CONF.

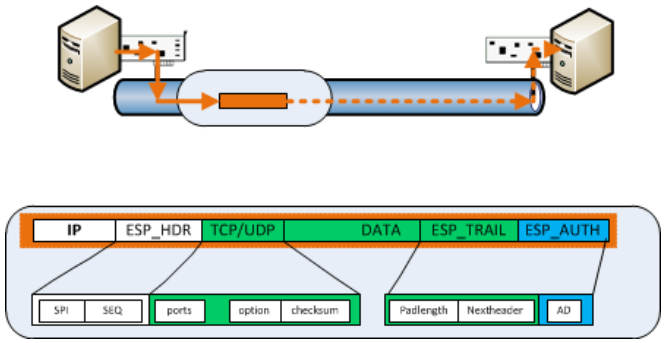

Figure 24. Datagramme IP protégé par ESP en mode transport

La Figure 25 représente les actions qui ont été appliquées au flux de protection en terme de confidentialité. On observe que tous les attributs de ESP Trail et du protocole du transport TCP/UDP et ce qui suit sont chiffrés par l'algorithme de sécurité DES3_ALGO. De plus, « ALL » signifie que l'attribut est complétement chiffré, son nom et sa valeur. Les attributs « NextHeader » et « PadLength » du protocole ESP et les attributs « PORTS » et « Checksum » du protocole TCP sont aussi totalement chiffrés par l'algorithme de sécurité DES3_ALGO et ainsi de suite.

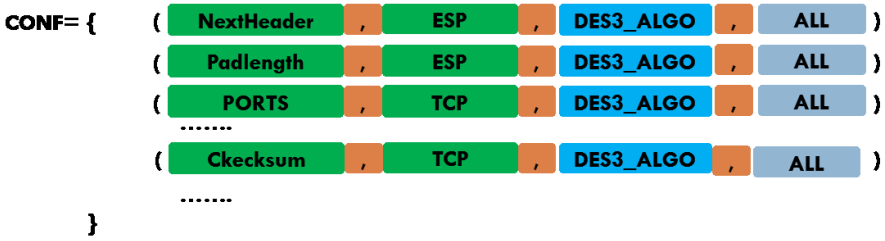

Figure 25. L'ensemble CONF

4.3 Modélisation des traitements sur les flux de données

Un traitement sur un flux de données est vu comme une fonction particulière de $\mathcal{F}$ (l'ensemble représentant les flux de données) dans $\mathcal{F}$. Lorsque ces fonctions sont combinées, elles doivent permettre de spécifier n'importe quel traitement d'un équipement de sécurité. Par conséquent, nous représentons un équipement non pas comme une entité propre mais par les traitements qu'il est capable d'appliquer sur les flux de données. Ces flux évoluent pendant leur traversée d'un réseau selon les mécanismes implantés sur les équipements du réseau. Cela peut être un protocole de chiffrement comme IPsec où une fonctionnalité transforme un flux de données en y ajoutant un service de sécurité (par exemple la confidentialité) et une autre fonctionnalité qui enlève cette transformation. Cela peut être également le NAT auquel cas une seule fonctionnalité de transformation est impliquée et aucun service de sécurité n'est ajouté au flux de données. Ainsi, chaque traitement considère un sous-ensemble des attributs du flux de données en entrée et va retourner le même flux de données, un nouveau flux de données ou rien si le flux est stoppé.

Dans [Laborde 2005], quatre familles de traitements sur un flux de données avaient été définis : la fonctionnalité de production/consommation, propagation, transformation ou filtrage. Nous proposons ici une approche différente où nous analysons les fonctions de base, que nous appelons **commandes primitives**, pouvant être appliquées sur notre modèle de flux de données. Ces commandes primitives permettront de composer des traitements plus évolués. Les équipements possèdent des capacités de traitement qui peuvent être un ensemble de fonctionnalités de base. Leurs caractéristiques spécifiques peuvent être différentes. Dans tous les cas on peut citer les changements suivants :

- *Ajouter des nouveaux blocs d'octets*. Par exemple, une passerelle IPsec ajoute l'entête correspondant au protocole AH entre le bloc IP et le bloc UDP/TCP quand ce protocole est utilisé en mode transport,
- *Supprimer des blocs d'octets*. Par exemple, un proxy http supprime le bloc IP lorsqu'il reçoit un flux de données, cela se retrouve dans les processus de dé-encapsulation.
- *Modifier des champs*. Par exemple, le NAT modifie la valeur du champ adresse IP source du bloc IP,
- *Authentifier des champs*. Par exemple, le protocole AH authentifie certains champs du protocole IP ainsi que les différents champs des protocoles encapsulés,
- *Chiffrer des champs*. Par exemple, le protocole ESP d'IPsec en mode transport chiffre les champs des protocoles encapsulés,
- *Rejeter tout le flux de données*. Par exemple, un pare-feu refuse un flux de données non-conforme à ses règles,
- etc.

Ajouté à cela, les mécanismes du réseau effectuent ces traitements par rapport à un sous-ensemble des différents champs constituant un flux de données. Par exemple, le NAT analyse simplement le champ adresse IP source ou destination selon le sens du flux de données. Un pare-feu sans état analyse uniquement les champs adresse IP source, adresse IP destination, protocole du bloc IP et les champs numéro de port source et destination du bloc UDP/TCP. Une passerelle analyse les mêmes champs. Il est donc nécessaire de fournir des commandes représenter l'accès aux informations dans le flux de données.

Nous présentons neuf commandes primitives. Deux permettent d'accéder à une donnée dans un flux, trois permettent de modifier un flux de données et quatre sont utilisées pour manipuler les ensembles AUTHN et CONF. L'idée est de spécifier des traitements simples de manière formelle afin d'éviter des erreurs dans la spécification de traitements plus complexes.

Nous utilisons la notation suivante dans la suite, pour obtenir un champ particulier d'un flux de données :

- Champ d'encapsulation: $f.\mathrm{e} \mid f \in \mathcal{F} \wedge \mathrm{e} \in \mathcal{E}$
- Champ d'authentification: $f.\mathrm{authn} \mid \mathrm{f} \in \mathrm{F} \wedge \mathrm{authn} \in \mathrm{AUTHN}$
- Champ de confidentialité: $f.\mathrm{conf} \mid \mathrm{f} \in \mathrm{F} \wedge \mathrm{conf} \in \mathrm{CONF}$
- Niveau de chiffrement :. $f.\mathrm{conf}.\ell \mid \mathrm{conf} \in \mathrm{CONF} \wedge \ell \in \mathcal{L}$
- Un protocole d'une encapsulation : $\mathrm{e.p} \mid \mathrm{p} \in \mathcal{P}$
- L'identifiant du protocole : $\mathrm{p.id} \mid$ qui représente l'identifiant du protocole de <name, id>
- Le nom d'un protocole: $\mathrm{p.protoName} \mid \mathrm{p} \in \mathcal{P}$ qui représente le nom du protocole de <name,id>
- les attributs d'un protocole: $\mathrm{p.attributes} \mid \mathrm{p} \in \mathcal{P} \wedge \mathrm{p.attributes} \subseteq \mathcal{A}$
- Le nom d'un attribut: $\mathrm{a.Name} \mid \mathrm{a} \in \mathcal{A}$ qui représente le nom d'un attribut.
- La valeur de l'attribut: $\mathrm{a.Value} \mid \mathrm{a} \in \mathcal{A}$ qui représente la valeur d'un attribut.

1) ***proto*** ← **Get_Protocol**(*f*, protoid)

// Retourne un protocole particulier du flux de donnée « *f* »

PRE

$\exists\, \mathrm{p} \mid \mathrm{p} \in f.\mathrm{e} \wedge \mathrm{p.id} = \mathrm{protoid}$

THEN

proto := p

ELSE

proto := null

END

END ;

2) ***attribute*** ← **Get_Attributes**(*f*, protoid, attName)

// Retourne l'attribut d'un protocole particulier s'il est lisible // i.e., s'il appartient à CONF, le niveau de chiffrement doit être = « VAL ».

PRE

$\exists\ \mathrm{p} = \mathrm{Get\_protocol}(f, \mathrm{protoid})$

$\exists\, \mathrm{a} \mid \mathrm{a} \in \mathrm{p.attributes} \wedge \mathrm{a.Name} = \mathrm{attName}$

$\forall(\mathrm{a}', \mathrm{p}', \mathrm{s}', \ell) \in \mathrm{f.CONF} \mid (\mathrm{a}' = \mathrm{a} \wedge \mathrm{p}' = \mathrm{p}) \Rightarrow \ell \neq \mathrm{all}$

THEN

attribute := a

ELSE

attribute := null

END

END;

3) ***flow*** ← **Modify_Attributes**(*f*, protoid, attribute)

// Modifie les attributs d'un protocole particulier. e.g., NAT change la valeur du champ de l'adresse IP source dans le bloc IP, ou un routeur change le champ ttl (time-to-live)

PRE

$\exists\ a = \text{Get\_Attributes}(f, \text{protoid}, \text{attribute.Name})$

$\exists\ p = \text{Get\_protocol}(f, \text{protoid})$

$\forall(a_1, p_1, a_2, p_2, s) \in f.\text{AUTHN} \mid$

$(a_1 \neq a \ \vee\ p_1 \neq p.\text{id}) \wedge (a_2 \neq a \ \vee p_2 \ \neq p.\text{id})$

THEN

$p'.\text{attribute} = (p.\text{attribute} - \{a\}) \cup \{\text{attribute}\}$

$f' =< e', \text{authn}, \text{conf} > \mid$

$e' = (e - \{p\}) \cup \{p'\}$

$flow := f'$

ELSE

flow := f

END

END;

4) ***flow*** ← **Add_Proto**(*f*, proto , protoid)

// Ajoute un nouveau protocole avant un protocole particulier. e.g., une passerelle IPsec ajoute l'entête AH entre le bloc IP et le bloc UDP/TCP (le protocole de transport) quand ce protocole est utilisé dans le mode de transport;
// « Proto » est le protocole à ajouter
// « Protoid » est l'identifiant du protocole qui doit succéder

PRE

$\exists\ p = \text{Get\_protocol}(f, \text{protoid})$

THEN

$f' =< e', \text{authn}, \text{conf} > \mid$

$e' = e \cup \{\text{proto}\}\ \text{where}\ p = \text{next}(\text{proto})$

$flow := f'$

ELSE

flow := f

END

END;

5) ***flow*** ← **Delete_Proto**(*f*, protoid)

// Supprime un protocole existant. e.g., un proxy HTTP supprime le bloc IP quand il reçoit un flux de données,

PRE

$\exists\ \mathrm{p} = \mathrm{Get\_Protocol}(f, \mathrm{protoid})$

$\forall(\mathrm{a}_1, \mathrm{p}_1, \mathrm{a}_2, \mathrm{p}_2, \mathrm{s}) \in f.AUTHN | (\mathrm{p}_1 \neq \mathrm{p.id} \wedge \mathrm{p}_2 \neq \mathrm{p.id})$

$\forall\ (\mathrm{a}', \mathrm{p}', \mathrm{s}', c)\ \in \mathrm{f.CONF} \mid (\mathrm{p}' \neq \mathrm{p.id})$

THEN

$f' =< \mathrm{e}', \mathrm{authn}, \mathrm{conf} > \mid \mathrm{e}' = \mathrm{e} - \{\mathrm{p}\}$

$flow := f'$

ELSE

flow := f

END

END;

6) ***flow*** ← **Add_AUTHN**(*f*, attribute1, protoid1, attribute2, protoid2, algo)

// Ajoute un élément dans AUTHN qui garantit l'intégrité d'un attribut donné. e.g., le protocole AH authentifie certains champs d'IP et tous les autres champs des protocoles encapsulés.

PRE

$\exists\ \mathrm{p} = \mathrm{Get\_Protocol}(f, \mathrm{protoid1}) \wedge \mathrm{p.id} = \mathrm{protoid1}$

$\exists\ \mathrm{q} = \mathrm{Get\_Protocol}(f, \mathrm{protoid2}) \wedge \mathrm{q.id} = \mathrm{protoid2}$

$\forall(\mathrm{a}_1, \mathrm{p}_1, \mathrm{a}_2, \mathrm{p}_2, \mathrm{s}) \in f.AUTHN | (\mathrm{p}_1 \neq \mathrm{p.id}\ \wedge \mathrm{p}_2 \neq \mathrm{q.id})$

THEN

$f' =$ e x authn′ x conf |

authn′ = authn ∪ {(attribute1, protoid1, attribute2, protoid2, algo)}

ELSE

flow := f

END

END;

7) ***flow*** ← **Delete_AUTHN**(*f*, attribute1, protoid1, attribute2, protoid2, algo)

// Supprime un élément existant dans AUTHN

PRE

$\exists\ \mathrm{p} = \mathrm{Get\_Protocol}(f, \mathrm{protoid1}) \wedge \mathrm{p.id} = \mathrm{protoid1}$

$\exists\ \mathrm{q} = \mathrm{Get\_Protocol}(f, \mathrm{protoid2}) \wedge \mathrm{q.id} = \mathrm{protoid2}$

$\forall(\mathrm{a}_1, \mathrm{p}_1, \mathrm{a}_2, \mathrm{p}_2, \mathrm{s}) \in f.\mathrm{AUTHN} | (\mathrm{p}_1 = \mathrm{p.id} \wedge \mathrm{p}_2 = \mathrm{q.id})$

THEN

$f' =$ e x authn′ x conf |

authn′ = authn - {(attribute1, protoid1, attribute2, protoid2, algo)}

ELSE

flow := f

END

END;

8) ***flow*** ← **Add_CONF**(*f*, attribute, protoid, algo, level)

// Ajoute un élément dans CONF

PRE

$\exists$ p = Get_Protocol(f, protoid) $\wedge$ p.id = protoid

$\forall$ (a′, p′, $\mathcal{s}'$, c) $\in$ f.CONF $\wedge$ p′ $\neq$ p.id

$\forall$ authn $\in$ f.AUTHN => authn $\neq \emptyset$

THEN

f' = e x authn x conf′ |

conf′ =conf $\cup$ {(attribute, protoid, algo, level)}

ELSE

flow := f

END

END;

9) ***flow*** ← **Delete_CONF**(*f*, attribute, protoid, algo, level)

// Supprime un élément existant dans CONF.

PRE

$\exists$ p = Get_Protocol(f, protoid) $\wedge$ p.id = protoid

$\forall$ (a′, p′, $\mathcal{s}'$, c) $\in$ f.CONF $\wedge$ p′ = p.id

THEN

f' = e x authn x conf′ |

conf′ = conf - {(attribute, protoid, algo, level)}

ELSE

flow := f

END

END;

4.4 Modèle de mécanisme abstrait basé sur l'attribut du flux

Maintenant que nous avons défini les commandes de base pour décrire des traitements évolués sur les flux de données, nous présentons une définition générique de mécanisme abstrait.

4.4.1 Définition formel d'un mécanisme

Un traitement sur un flux de données est une fonction particulière de $\mathcal{F}$ dans $\mathcal{F}$. Ce traitement est effectué par un mécanisme spécifique avec une configuration spécifique. Un mécanisme spécifique a une *capacité* donnée qui représente ce que le mécanisme peut faire. Par exemple, un pare-feu peut filtrer les datagrammes IP en analysant les adresses IP, ports, etc. Un proxy HTTP peuvent modifier les messages HTTP en analysant un ensemble de champs HTTP.

La deuxième composante d'un mécanisme spécifique est sa *configuration*. La configuration définit un comportement particulier sur la base de la capacité du mécanisme. Prenons l'exemple d'une configuration d'un firewall : « si l'adresse IP source est égale à 1.2.3.4 et le port source TCP est inférieur à 1024 alors bloquer ». Cette configuration demande au firewall d'avoir les capacités suivantes :

1) récupérer l'adresse IP de source et le port source TCP du datagramme IP,

2) exécuter les fonctions « adresse IP est égal à » et « le port est inférieur à » et

3) appliquer l'action « rejeté ».

En conséquence, nous définissons un mécanisme « M » par sa capacité propre, notée CAPABILITY_M, et par sa configuration, notée CONFIGURATION_M, qui va définir son comportement par rapport à sa capacité :

$$\mathbf{M = < CAPABILITY_M, CONFIGURATION_M >}$$

4.4.2 Définition de la capacité d'un mécanisme

On désigne par $\boldsymbol{A_M^F}$ l'ensemble des attributs d'un flux de données F (Définition 1) qui peuvent être observés par un mécanisme *M*, et par $\boldsymbol{A_M^{ctx}}$ l'ensemble des attributs de contexte trouvés dans un règle de configuration du mécanisme *M*. Un attribut de contexte est un attribut qui ne se trouve pas dans le flux de données mais qui peut être utilisé par le mécanisme *M* pour effectuer un traitement (e.g., - o eth0 (- out interface) ou les informations sur l'état des connexions dans un règle iptables). Chaque attribut a un type de données défini qui appartient à Σ_M, l'ensemble non-vide des types de données que *M* reconnaît (e.g., l'attribut @ips a le type de données IP_Address).

Les notations qui suivent nous permettent de représenter la capacité d'un mécanisme :

- $Type_M(a_i)$ le type des attributs a_i où $a_i \in A_M^F \cup A_M^{ctx}$. $Type_M(a_i) \in \Sigma_M$.
  - Exemple: IP_Address = $Type_M$(@ips) ou STRING = $Type_M$(protoname) etc.
- $Value(a_i)$ la valeur d'attribut a_i.
  - Exemple: Value(@ips) = 10.2.1.11
- $Values(a_i)$ l'ensemble des valeurs des attributs a_i.
  - Exemple :Values(@ips) = 1.0.0.0/16 (range of IP Adresses) ; Values(ports) = [1..1024] (Public port numbers, etc.)

On désigne par $EXPR_M$ l'ensemble des expressions fournies par un mécanisme de sécurité M. L'ensemble de tous les attributs dans une expression « e_M » est définie par $Attr(e_M)$.

Définition 3. ***Définition formelle du*** $\mathbf{CAPABILITY_M}$

$\mathbf{CAPABILITY_M} = < \mathbf{\Sigma_M}, \mathbf{A_M}, \mathbf{EXPR_M^A} >$ où :

1) $\boldsymbol{\Sigma_M}$ est l'ensemble fini non-vide des types de données reconnus par *M* – l'ensemble des flux de données appartient à $\boldsymbol{\Sigma_M}$ donc $\mathcal{F} \in \Sigma_M$;
2) $\boldsymbol{A_M = A_M^F \cup A_M^{ctx}}$ | A_M={a_i | 1 ≤ i ≤ k ou $\mathbf{k} \in \mathbb{N}$ et $Type(a_i) \in \Sigma_M$ };
3) $\boldsymbol{EXPR_M^A}$ est l'ensemble fini des expressions évaluables par *M* et portant sur des *variables libres* appartenant à A_M ;

4.4.3 Définition formel de la configuration d'un mécanisme

Basés sur la notion de capacité définie précédemment, nous définissons les éléments de configuration comme suit:

Définition 4. *Définition formelle de la configuration :* $\boldsymbol{CONFIGURATION_M}$

Une configuration $\mathbf{c} \in \mathbf{CONFIGURATION_M^A}$ d'un mécanisme M est une liste de règles « $\mathbf{RULE_M^A}$ » (Définition 5) et un algorithme de résolution des conflits « CRA » (cf. section 4.4.4). Les règles définissent le comportement du mécanisme alors que l'algorithme permet au mécanisme de déterminer quelle règle il faut appliquer lorsque plusieurs peuvent être exécutées.

A la fois les règles de configurations ainsi que l'algorithme de résolution de conflit dépendent de la capacité du mécanisme M, « $CAPABILITY_M$ » (Définition 3):

$$\mathbf{CONFIGURATION_M} \in \mathbf{RULES_M} \times \mathbf{CRA_M}$$

Définition 5. *Définition formelle d'une règle:* $\boldsymbol{RULE_M^A}$

Une règle d'un mécanisme M consiste en un ensemble de contraintes sur l'ensemble des attributs A_M que le mécanisme M reconnaît et un ensemble d'une ou plusieurs actions construit(es) par rapport aux commandes primitives.

Chaque règle $r \in RULE_M^A$ est de la forme si *condition* alors *action* et donc :

$$\mathbf{RULE_M^A} \subseteq \mathbf{CONDITION_M^A} \times \boldsymbol{ACTION_M^A}$$

Définition 6. *Définition formelle d'une condition :* $\boldsymbol{CONDITION_M^A}$

$CONDITION_M^A$ est l'ensemble des expressions booléennes basées sur « A_M » que le mécanisme M peut évaluer. Par conséquent, il est défini ainsi :

$CONDITION_M^A \subseteq EXPR_M^A | \forall cond_M \in CONDITION_M^A$ où Type($cond_M$)=Bool et Type(Attr($cond_M$)) $\subseteq \Sigma_M$

Une condition peut être représentée sous la forme normale conjonctive, i.e., $[cond_M] = \{cond_{a_1}, cond_{a_2}, \dots, cond_{a_i} | i \in \mathbb{N}\} = \bigwedge_{i \in N}[cond_{a_i}]$. Si tous les éléments de « $cond_a$ » sont vrais, [$cond_M$] est évaluée à vrai. Par contre, si un des éléments est faux, alors [$cond_M$] échoue et la règle suivante sera testée.

$cond_M \rightarrow a_i$ rel_op a_j

| a_i rel_op Value(a_j)

| a_i set_op Values(a_i)

| ! $cond_M$

| $cond_M$ and $cond_M$

| $cond_M$ or $cond_M$

| ($cond_M$)

rel_op $\rightarrow = | \neq | < | \leq | > | \geq$

set_op $\rightarrow \in | \notin | \subset | \subseteq | \not\subset$

Définition 7. ***Définition formelle d'une action :*** $\boldsymbol{ACTION_M^A}$

L'ensemble $ACTION_M^A$ correspond à l'ensemble des traitements que le mécanisme M peut effectuer sur un flux de données (modification, blocage, etc.). Les actions sont construites sur les commandes primitives définies plus haut. Ainsi, une action est une expression dont le type de données est $\mathcal{F}$:

$ACTION_M^A \subseteq EXPR_M^A | \forall$ action $\in ACTION_M^A$, Type(action) = $\mathcal{F}$

4.4.4 Algorithme de résolution de conflits CRA_M

Les mécanismes de sécurité comportent divers algorithmes de résolution de conflit afin de rendre déterministe le processus de prise de décision sur un flux de données comme « deny-takes-precedence », « first-match-takes-precedence », « more-specific-takes-precedence », etc. De plus selon les mécanismes, il peut y avoir un ou plusieurs algorithmes implantés.

Notre approche se voulant au maximum générique, il nous faut représenter cette diversité de manière unifiée. La résolution des conflits peut devenir complexe en raison de l'existence de hiérarchies d'héritages sophistiquées comme l'héritage de rôles pour des systèmes d'autorisation de type RBAC mais aussi de la diversité des façons de combiner les politiques de résolution. Afin de fournir de représentation unifiée des algorithmes de résolution de conflits, nous réutilisons le travail de Chinaei et al. [Chinaei et al. 2010]. Les auteurs ont proposé un cadre unifié avec un seul algorithme paramétrable qui prend en charge toutes les combinaisons légitimes, et ce en utilisant uniquement quatre politiques de résolution de conflits (Figure 26) : autorisation préférée, localité, majoritaire, et l'autorisation par défaut.

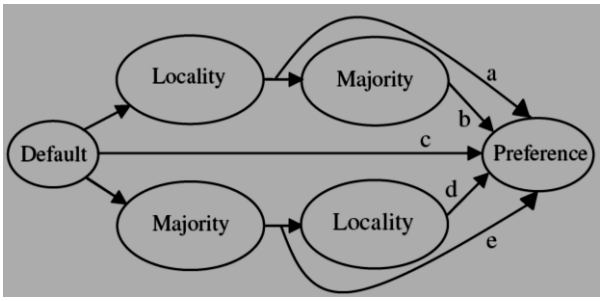

Figure 26. Stratégie de résolution des conflits

4.5 Représentation de notre modélisation dans les réseaux de Petri colorés hiérarchiques

Afin de pouvoir utiliser cette modélisation pour analyser des conflits de configurations de sécurité réseau, nous avons exprimé tous nos concepts dans le formalisme des réseaux de Petri colorés hiérarchiques. Cet exercice nous permet d'automatiser l'analyse de conflits de sécurité réseau grâce aux outils associés et notamment CPN tools. Ce travail est présenté dans cette section.

4.5.1 Introduction aux réseaux de Petri hiérarchiques

Les réseaux de Petri Colorés (CPNs : Coloured Petri Nets) [Jensen et Kristensen 2009] sont un langage de spécification formel (il possède une définition mathématique de sa syntaxe et sa sémantique) pour construire des modèles de systèmes concurrents et analyser leurs propriétés. Ce langage est basé sur les réseaux de Petri ordinaires et fait donc partie des langages de modélisation à évènements discrets. Un réseau de Pétri ordinaire est un graphe biparti où les nœuds sont appelés

places (représentées par des cercles) ou transitions (représentées par des rectangles). Les nœuds sont connectés par des arcs valués. La valuation des arcs en entrée des transitions indique la condition d'activation des transitions, tandis que la valuation des arcs en sortie des transitions reflète l'effet de l'action de la transition. Chaque place peut contenir une ou plusieurs marques, aussi appelés jetons. L'ensemble des jetons contenus dans l'ensemble des places d'un réseau de Petri ordinaire définit son état. Le marquage initial de l'ensemble des places d'un réseau de Petri ordinaire représente son état initial. Le marquage d'un réseau de Petri ordinaire, et donc son état, évolue à chaque activation d'une transition. Une transition *t* ne peut être activée que si le marquage de l'ensemble de places connectées par un arc entrant correspond à la valuation de l'arc. A l'activation de la transition, un nombre de jetons correspondant à la valuation des arcs entrants sont consommés des places entrantes. Des jetons sont aussi produits dans les places sortantes conformément à la valuation d'arcs sortants. L'analyse d'un réseau de Petri ordinaire peut se faire soit par simulation, soit par l'analyse du graphe d'état.

Les CPNs fournissent en plus les capacités des langages de programmation de haut niveau permettant d'améliorer l'expressivité de ce langage. Les jetons peuvent avoir différentes valeurs, appelées couleurs, correspondant au domaine de couleurs des places dans lesquelles ils se trouvent. Un domaine de couleur correspond à un type de données. Le langage de programmation CPN ML [Harper 2011], qui est basé sur le langage de programmation fonctionnel Standard ML, fournit des primitives pour définir des types de données, pour décrire la manipulation des données et ainsi créer des modèles compacts et paramétrables. Cela permet de définir des filtres puissants au niveau des transitions mais aussi associer aux arcs sortants des traitements évolués sur les jetons.

Par exemple, la Figure 27 présente un CPN avec deux états (à gauche et droite), deux domaines de couleurs de jetons, trois places et une transition. Dans le premier état à gauche du système, on remarque les places '1' et '2' contiennent chacune un jeton en mauve (de type <*entier,string*>) et en jaune (de type <*entier*>). Pour que la transition soit activée, il faut que la place 1 et la place 2 contiennent au moins un jeton chacune. Il n'y a pas de restriction sur les valeurs de jetons. Lorsque la transition est tirée, le système passe au deuxième état à droite où les places '1' et '2' deviennent vides (consommation) et la place '3' contient un jeton en vert (de type <*entier,string*>). Le nombre et la valeur du/des jeton(s) produit(s) dépendent de l'arc sortant. Ici, il s'agit de multiplier le premier champ du jeton violet avec la valeur du jeton jaune et de recopier la valeur du deuxième champ du jeton violet, c'est-à-dire <4 * 2, « Foo »>.

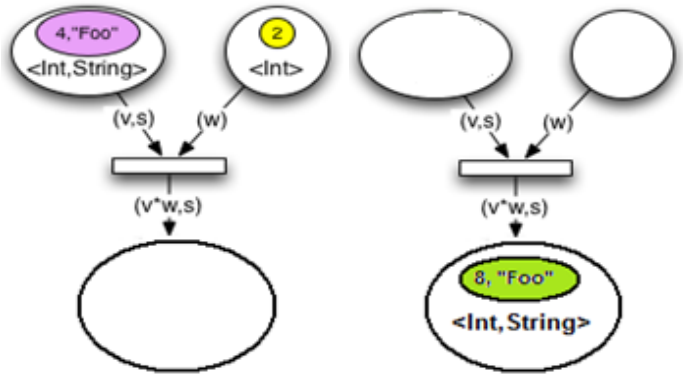

Figure 27. Exemple de graphe d'état d'un réseau de Petri coloré

La définition formelle d'un CPN est un 9-tuple CPN=(P, T, A, Σ, V, C, G, E, I) où

- P est l'ensemble des places
- T est l'ensemble des transitions
- $A \subseteq P \times T \cup T \times P$ est l'ensemble des arcs
- Σ est l'ensemble des domaines de couleurs
- V est l'ensemble des variables avec $Type[v] \in \Sigma$ pour tout $v \in V$
- $C : P \rightarrow \Sigma$ est une fonction qui associe une couleur à chaque place
- $G : T \rightarrow EXPR_V$ représente les fonctions de garde positionnées sur les transitions donc Type[G(t)] = Bool où pour tout $t \in T$
- $E : A \rightarrow EXPR_V$ est fonction qui associe une expression à chaque arc. Type[E(a)] = C(p) pour tout $a \in A$, où p est la place connectée à l'arc.
- $I : P \rightarrow EXPR_{\emptyset}$ est la fonction d'initialisation.

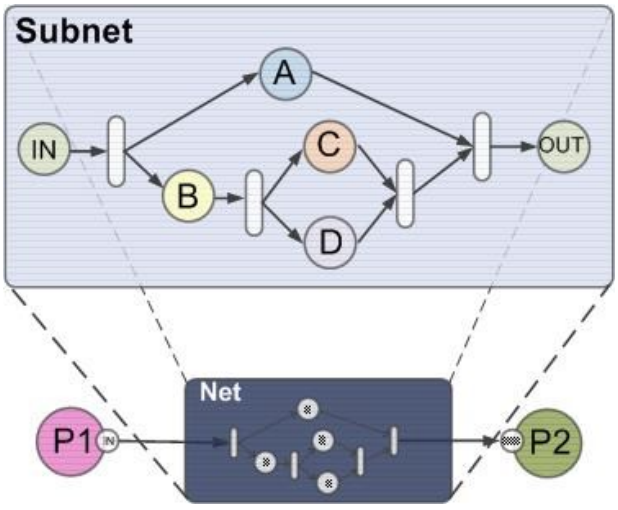

Figure 28. CPN hiérarchique

Ajouté à cela, il existe une extension des CPNs appelée CPNs hiérarchiques qui permet de structurer différents niveaux d'abstraction. Un réseau de Petri hiérarchique est un modèle dans lequel une partie est représentée par une transition de substitution. Ainsi, il est possible de subdiviser un modèle en différentes parties permettant une modélisation modulaire. La Figure 28 montre un exemple d'une telle hiérarchie. La transition bleu foncé est une transition de substitution qui correspond au réseau de Pétri dans le cadre bleu ciel. Il est nécessaire que le domaine de couleur de la place P1 (respectivement P2) corresponde à celui de la place IN (respectivement OUT). Il alors possible d'améliorer la lisibilité d'une spécification et faisant abstraction de certains détails pour se concentrer à un niveau d'abstraction élevé uniquement.

Nous avons donc choisi de représenter notre modélisation de mécanismes de sécurité dans le formalisme des réseaux de Petri colorés hiérarchiques [Jensen et Kristensen 2009] pour les raisons suivantes :

- Les réseaux de Pétri ordinaires facilitent la spécification et l'analyse traitant de problèmes de concurrence, de synchronisation, de partage de ressource et de parallélisme,
- Les réseaux de Pétri colorés permettent de spécifier des types de données et des traitements évolués. En particulier, Laborde et al. [Laborde et al. 2007] ont utilisé ce formalisme pour définir un flux de données selon leur modèle,
- Les réseaux de Pétri hiérarchiques autorisent une modélisation modulaire,
- Il existe le logiciel CPN tools [CPNTools] qui fournit un environnement complet pour spécifier des CPN hiérarchiques et les analyser par simulation ou par étude du graphe d'état.

4.5.2 Définition des couleurs et fonctions utilisées

Nous avons utilisé le logiciel « CPN Tools » [CPNTools] dans le cadre de nos travaux pour créer et simuler nos CPN. Cet environnement de développement de CPN utilise une extension du langage ML [Harper 2011] pour spécifier par exemple les couleurs de jeton, les gardes, les fonctions sur les arcs. Le langage CPN ML nous a permis de définir des couleurs pour représenter le type flux de données, ainsi que le type de données Mécanisme correspondant à sa capacité et une configuration. La Figure 29 présente les définitions ML des flux de données (cf. Définition 1). Par conséquent, un flux de données sera représenté par un jeton du domaine de couleur DATAFLOW, une configuration sera aussi un jeton, etc.

Nous avons aussi écrit en CPN ML toutes les commandes primitives. Ces commandes permettent par la suite d'écrire des fonctions particulières sur les flux de données comme un

traitement correspondant à la mise œuvre du protocole IPsec AH en mode transport. Les commandes primitives en CPN ML sont présentées dans la Figure 30.

```
DATAFLOW
  colset ATTRIBUTE = record name: STRING * value: VALUES;
  colset PROTOCOLID = record name: STRING * id: INT;
  colset ATTList = list ATTRIBUTE;
  colset PROTOCOL = record protoid: PROTOCOLID * attributes: ATTList;
  colset ENCAPSULATION = list PROTOCOL;
  colset SECALGO = with DES | TDES | HMAC;
  colset AUTHN = product ATTRIBUTE * PROTOCOLID  *
  ATTRIBUTE *PROTOCOLID  * SECALGO;
  colset AUTHNLIST = list AUTHN;
  colset LEVEL = with ALL | VAL ;
  colset CONF = product ATTRIBUTE * PROTOCOLID  * SECALGO * LEVEL;
  colset CONFLIST = list CONF;
  colset DATAFLOW = product ENCAPSULATION * AUTHNLIST * CONFLIST;
```

Figure 29. Définition du flux de données en CPN-ML

```
GENERALISATION
  fun Get_Protocol(e:ENCAPSULATION, pid:PROTOCOLID)=
  (*return a protocol from a dataflow if the protocolid exist*)
  if e=nil
   then {protoid={name="",id=0},attributes=[]}
  else  if (#protoid (hd e)) = pid
        then hd e
        else Get_Protocol(tl e,pid)
  fun Get_Attribute
  fun Add_att
  fun Del_att
  fun Add_Proto(d:DATAFLOW,p:PROTOCOL,pid:PROTOCOLID)=
  (add(#1 d,p,(#protoid (Get_Protocol(#1 d,pid)))),#2 d,#3 d)
  fun Delete_Proto
  fun Add_AUTHN
  fun Delete_AUTHN
  fun Delete_CONF
  fun Add_CONF
  fun Modify
  fun Modify_Attribute
```

Figure 30. Définition des commandes primitives en CPN-ML

4.5.3 Un modèle CPN générique pour représenter des mécanismes

Les réseaux de Petri colorés hiérarchiques favorisent des spécifications modulaires des transitions dites de substitution correspondront à un CPN hiérarchique particulier. Afin de permettre la réutilisabilité des modèles CPN, nous proposons de définir un modèle CPN unique pour représenter tout mécanisme de sécurité. Nous appelons GAM (Generic Attribute-based Mechanism) ce modèle CPN. Ainsi, un mécanisme particulier pourra être construit comme une transition de substitution

correspondant à un GAM qui aura été spécialisé uniquement par un jeton décrivant ce mécanisme, c'est-à-dire sa capacité et sa configuration (jeton du domaine de couleur M).

La Figure 31 est le modèle CPN du GAM. Le GAM prend en entrée des jetons de type DATAFLOW (place F marqué par « In » en bleu) et retourne des jetons de type DATAFLOW comme sortie (place F' marqué par « Out » en bleu).

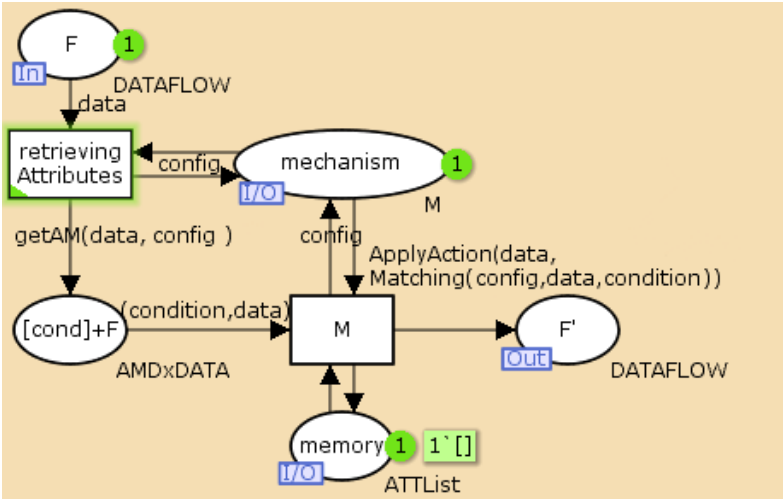

Figure 31. Présentation de GAM en CPN

La spécialisation de mécanisme générique effectuée dans la place « mechanism » qui contient la capacité et la configuration de ce mécanisme (cf. section 4.4, Définition 3 et Définition 44.4). Cette place est notée in/out un mécanisme donné devra comporter une place correspondante. De même, les attributs contextuels trouvés dans une règle de configuration du mécanisme M « A_M^{ctx} » sont stockés dans la place « memory » (cf. section 4.4.2). Cette mémoire peut être utilisée pour représenter des mécanismes de stateful [frozentux] par exemple. De plus, la place « memory » peut être partagée par différents mécanismes.

Informellement, le comportement de GAM-CPN est expliqué par l'algorithme suivant :

Entré: (DATAFLOW, M)

Sortie: (DATAFLOW)

étape 1: getAM(DATAFLOW, M)

Cette fonction récupère les attributs A_M observables par le mécanisme à partir du flux de données (condition variable)

étape 2: Matching(DATAFLOW, M, AMD)

2.1 Cette fonction retourne toutes les règles qui correspondent aux attributs récupérés dans le flux de données et la mémoire.

2.2 S'il y a au plus une règle de correspondance dans la liste « match », aller à l'étape 4.

étape 3: CRA(DATAFLOW, RULES, CONFIGURATION)

À ce point, la liste assortie contient au moins deux règles (obtient via étape 2). CRA (cf. section 4.4.4) sera appliqué afin de réorganiser et supprimer des règles qui ne doivent pas être appliquées, puis passez à l'étape 4.

étape 4: ApplyAction(DATAFLOW, RULES)

Actions de règles de correspondance sont appliquées. Le résultat est un flux de données qui pourrait être vide, inchangé ou modifié.

Figure 32. Algorithme de GAM-CPN

Le mécanisme particulier est spécifié une transition de substitution faisant référence à un GAM. La spécialisation du GAM en ce mécanisme est effectuée en créant un jeton décrivant le mécanisme. Globalement, ce jeton correspond à la configuration du GAM pour qu'il se comporte comme le mécanisme cible. La Figure 33 propose un exemple de spécialisation. Il s'agit ici de représenter deux traitements appliqués sur un flux (IPsec, puis IPtables). Les transitions BlackBox1 et BlackBox2 sont des transitions de substitution vers deux GAMs (les deux GAMs sont exactement les mêmes). La spécialisation pour représenter le comportement d'IPsec (respectivement IPtables) correspond au

jeton vert dans la place IPsec (respectivement IPtables). Notre approche de description facilite ainsi la lisibilité des spécifications.

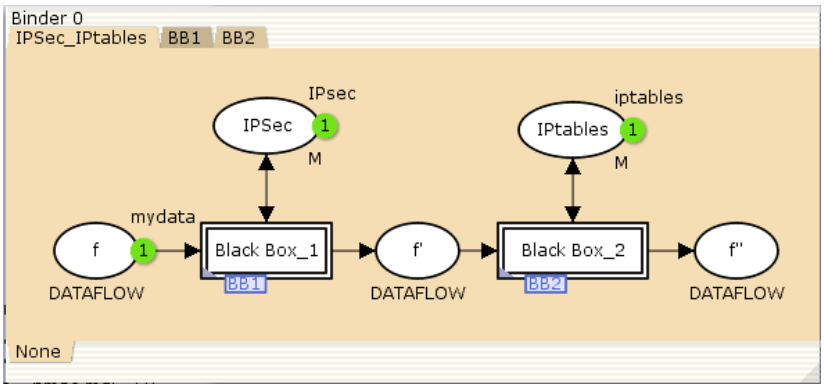

Figure 33. Exemple de spécialisation du GAM

4.6 Conclusion

La sécurité des réseaux dépend de la coordination de différents dispositifs hétérogènes (multi-mécanismes et multi-niveaux en terme de couche ISO) et interdépendants. De plus, l'évolution rapide des technologies impose que la méthode d'analyse de la sécurité soit indépendante de la technologie utilisée. Nous avons proposé dans ce chapitre une modélisation des mécanismes réseau orienté flux de données ([El-Khoury et al. 2011 b] et [El-Khoury et al. 2012 a]). Ce modèle est indépendant des mécanismes de sécurité réels pour prendre en compte leur diversité ainsi que leur évolution possible.

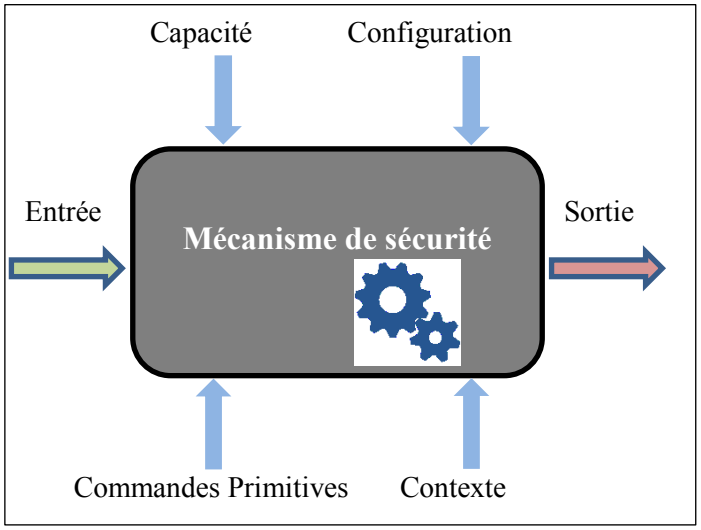

Figure 34. Mécanisme générique de sécurité

Nous avons dans un premier temps élaboré un modèle de flux de données en se basant sur le concept d'encapsulation de protocoles ainsi que sur l'approche de visualisation offerts par les outils d'analyse réseau. Notre représentation formelle a été basée sur une abstraction d'un flux de données physique constitué de blocs protocole ; chaque protocole étant constitué d'attributs. Nous avons aussi

ajouté l'historisation des traitements de sécurité d'authentification et de confidentialité afin d'obtenir une vision précise du flux ainsi que de sa protection.

Basé sur cette modélisation, nous avons pu proposer une représentation abstraite d'un mécanisme de sécurité générant un flux de données en sortie par rapport à un flux de données en entrée (Figure 34). De manière informelle, un mécanisme de sécurité est caractérisé par une capacité englobant l'ensemble des informations qu'il est capable de collecter pour prendre une décision ainsi qu'une ensemble de fonctions d'évaluation et de traitement. Ces fonctions sont créées par rapport à des commandes primitives qui ont été formellement décrites. Un mécanisme possède aussi une configuration qui va contrôler son comportement par rapport à ses capacités. Les algorithmes de résolution de conflits peuvent aussi être spécifiés de manière uniforme.

Afin de pouvoir utiliser cette modélisation pour analyser des conflits de configurations de sécurité réseau, nous avons exprimé ces concepts dans le formalisme des réseaux de Petri colorés hiérarchiques. Nous avons pu conserver dans ce formalisme notre représentation abstraite d'un mécanisme grâce au modèle CPN-GAM que nous avons proposé. Un mécanisme particulier sera un GAM avec un jeton décrivant sa capacité et sa configuration. Ce travail a été effectué dans l'outil CPN Tools ce qui nous d'analyser via la simulation des conflits de configurations de sécurité réseau.

Le prochain chapitre présente des cas d'étude qui montrent les capacités d'expression et d'analyse de notre travail.

Chapitre 5. Etudes de cas : Spécification et analyse de configurations de sécurité réseau

5.1 Introduction

Ce chapitre a pour objectif de donner une illustration des fondements que nous avons proposés dans les chapitres précédents. Les protocoles de la famille TCP/IP sont très largement exploités que ce soit dans le cœur des réseaux de communication ou encore dans la réalisation des connexions de postes de travail au sein des réseaux d'accès ou d'extrémité. A l'origine cette pile de protocole de communication avait pour objectif de garantir l'acheminement des unités de données entre des systèmes d'extrémité dont le nombre était restreint. Le nombre de réseaux traversés était considérablement limitée par voie de conséquence il n'y avait de complexité naissant de la pluralité des entités administratives de gestion des réseaux. Mais le succès de cette technologie d'interconnexion de système s'est accompagné par l'émergence de problématiques dont la nécessité de supporter la connexion de milliers de systèmes, la nécessité de traverser de multiples réseaux gérés par des entités administratives différentes, concurrentes, la nécessité d'exploiter des mécanismes de sécurisation des échanges à différents niveaux pour protéger les informations transmises et les systèmes cibles.

La croissance ininterrompue du nombre de systèmes raccordés au réseau a conduit très rapidement à la pénurie d'adresse. En effet le choix de coder à l'origine sur 4 octets, les adresses de systèmes raccordés au réseau a considérablement réduit les possibilités de raccordement physique des systèmes. Très rapidement, la technique de translation d'adresses (NAT en anglais, [RFC 3022]) est alors devenue une pratique courante pour pallier au manque croissant d'adresses IPv4 libres. La publication de la [RFC 1918] a permis de définir des plans d'adressage de systèmes privatifs. Les adresses appartenant à ces plans d'adressage ne sont pas routables sur Internet et ne doivent pas être utilisées par des machines de ce réseau. En conséquence, la translation d'adresse est utilisée pour permettre aux machines du réseau privé d'accéder à Internet, et de façon plus générale à d'autres réseaux. La substitution « adresse publique—adresse privée » est utilisée pour qu'un système appartenant à un adressage privé puisse atteindre soit un système figurant dans un réseau public, soit figurant dans un réseau privé (à condition d'opérer dans ce dernier cas une nouvelle substitution). La

mise en place de cette conversion permet de préserver et de limiter le nombre d'adresses publiques. Cependant afin de protéger les systèmes qui engagent des opérations de transfert de données, il est nécessaire de définir et d'exécuter des politiques de sécurité pour contrôler la conduite de ces échanges. L'utilisation d'une adresse ou d'un pool d'adresses au sein de ces équipements peut alors donner lieu soit à des situations de blocages ou bien d'ouverture d'accès extérieurs non sollicités et être source de problèmes.

L'extension IPsec (Internet Protocol Security) est une suite de protocoles normalisés par l'IETF qui fournit des services de sécurisation des données au niveau de la couche réseau. Les [RFC 4301 à 4309] décrivent le mode opératoire de cette extension (qui présente l'avantage d'être à la fois commun aux normes Ipv4 et Ipv6). En raison de la mise en œuvre de communications entre des équipements distants, séparés par la traversée de réseaux pour lesquels il n'est pas possible de garantir la maîtrise de toute opération réalisée par les équipements intermédiaires traversés, cette extension apporte les propriétés suivantes :

- **Confidentialité :** en rendant impossible l'interprétation de données si l'on en est pas le destinataire ; les techniques de chiffrement assurent la transformation des données intelligibles (en clair) en données inintelligibles (chiffrées).
- **authentification :** offre l'assurance qu'une donnée provient bien de l'origine de laquelle elle est censée provenir.
- **intégrité :** service qui consiste à s'assurer qu'une donnée n'a pas été altérée accidentellement ou frauduleusement
- **protection contre le rejeu :** permet d'empêcher les attaques consistant à renvoyer des données valides précédemment communiquées sur le réseau pour obtenir les mêmes accréditations que celle accordées légitimement au cours de l'échange précédant.
- **gestion des clés :** mécanisme de négociation de la longueur des clés de chiffrement entre deux éléments IPsec et d'échange de ces clés.

La mise en œuvre d'IPsec donne lieu à des décisions stratégiques qui relèvent également de choix effectués et/ou impactés par la conception et l'exécution d'une politique de sécurité. Comme nous l'avons indiqué dans les chapitres précédents, une communication entre deux systèmes peut être analysée comme l'acheminement d'un flux « affecté ou transformé » par une fonctionnalité mise en œuvre au sein d'un équipement. Les règles de précédences dans la réalisation de ces opérations, leur compatibilité ou incompatibilité peut être source de problèmes et conduire à des situations de blocage. Notre travail a pour objectif de pouvoir en déceler la teneur.

Ceux qui ont expérimenté la conjugaison de la fonctionnalité de translation d'adresse réseau et de la mise en œuvre du protocole IPsec ont inévitablement rencontré des écueils car la cohabitation entre la fonctionnalité NAT et IPsec est difficile. La fonctionnalité NAT modifie les paquets IP. La

conséquence directe est que tout contrôle d'intégrité au niveau IP et même aux niveaux supérieurs est cassé puisque TCP par exemple inclue les adresses dans ses checksums! Concrètement, on se rend compte qu'un protocole de sécurisation des datagrammes comme IPsec est incompatible avec le NAT, que ce soit en mode tunnel ou transport. Une autre raison simple est qu'un NAT évolué a tendance à remonter les couches pour étudier les protocoles de transport afin de rassembler assez d'informations pour chaque contexte. Tout chiffrement à ce niveau empêcherait donc le NAT de fonctionner, puisque les informations seraient alors chiffrées.

Dans ce chapitre, nous illustrons la possibilité de détecter des conflits entre technologies de sécurité sans avoir de connaissance ou d'expérience relatives à ce domaine. Pour cela nous nous appuyons sur des exemples présentant des usages à la fois simples et complexes. L'existence de l'incompatibilité entre IPsec et NA(P)T est établie depuis longtemps [RFC 3715]. Il n'est cependant pas trivial d'en déceler la teneur sans l'aide d'une expertise humaine. Nous nous appuyons tour à tour sur des exemples liés à l'utilisation d'IPsec [RFC 4301] à savoir la mise en œuvre des protocoles AH (Authentication Header), ESP (Encapsulating Security Payload), et du NA(P)T (Network address and port translation) pour démontrer la faisabilité de notre approche car en effet, notre approche n'est pas liée aux technologies décrites et peut être appliquée à d'autres problèmes de la même façon. La mise en œuvre de firewall offre des niveaux de complexité plus ou moins forts. La technologie des « iptables » donne lieu à l'exploitation de plusieurs mécanismes (MANGLE, FILTER, NAT, etc.) pour définir le traitement réservé à un flux traversant un firewall de ce type. La complexité d'une telle solution de firewall permet d'éprouver notre solution en proposant une analyse rigoureuse de la détection de conflits. Par ailleurs elle ouvre des possibilités d'investigation en s'attachant à l'observation et à l'influence que peuvent avoir des changements de valeur d'attributs affectant un flux.

Dans le chapitre précédent nous avons affiché les notations et les outils utilisés pour arriver à appliquer les scénarii que nous proposons dans ce chapitre.

La deuxième section de ce chapitre rappelle les spécifications du protocole IPsec et l'exploitation de notre formalisme sur les protocoles AH et ESP. La troisième section rappelle les fonctionnalités de la translation d'adresse réseau (NAT) et l'exploitation de notre formalisme sur (NA(P)T). La quatrième section présente les aspects principaux des firewalls, puis l'architecture de la technologie iptables et plus particulièrement l'intérêt de l'exploitation de mangle. A la fin de chaque section un exemple de mise en œuvre des technologies choisies est présenté.

Le modèle GAM (Generic Attribute-based Mechanism) a été introduit dans le chapitre précédent. Il offre une modélisation des fonctionnalités mises en œuvre au cours d'une communication. Ce modèle permet tout aussi bien d'approcher les technologies sous une forme macroscopique, donnant la possibilité de les exploiter sous la forme de « boites noires », tout comme ce modèle permet d'étudier finement le fonctionnement d'une fonctionnalité élémentaire. Chaque « boite » traite un flux entrant avant de fournir en sortie un nouveau flux. La cinquième section est

consacrée à la présentation dans un premier temps de trois scénarii, illustrant l'utilisation des ensembles « AUTHN et CONF » (cf chapitre 4, section 4.3, Définition 1) ainsi que la mise en œuvre de tunnels en s'appuyant sur notre formalisme. Un quatrième scénario relatif à la mise en œuvre de fonctionnalités « iptables » est également proposé afin de montrer la possibilité de détecter des anomalies qui peuvent naître de l'utilisation conjointe de mécanismes propres à la technologie iptables (intra-iptables).

5.2 Rappels relatifs au protocole IPsec

Les services de sécurité fournis par IPsec reposent sur la mise en œuvre de l'un des deux protocoles AH (Authentication Header) et ESP (Encapsulating Security Payload) qui constituent le cœur de la technologie IPsec [RFC 2401] et [RFC 4301]. Ces deux protocoles transforment différemment les datagrammes IP dans l'adressage des problématiques de sécurité.

Le protocole AH [RFC 4302] traite principalement la problématique de l'intégrité : en ajoutant un nouveau champ AH au datagramme initial, il permet d'une part de s'assurer que les paquets échangés n'ont pas été altérés et d'autre part de garantir l'identité de l'expéditeur d'un paquet. Il garantit aussi une protection contre le rejeu. Cependant AH ne s'attache pas à la problématique de la confidentialité des données puisque les informations contenues dans le datagramme initial ne subissent aucune transformation.

Le protocole ESP [RFC 4306] quant à lui traite de la confidentialité, l'authentification des données échangées. Il traite aussi de l'intégrité et garantit aussi une protection contre le rejeu. Il est possible d'utiliser uniquement les fonctions d'intégrité et d'authentification sans chiffrement et dans ce contexte le cas d'usage de AH peut être mis en cause.

Ces deux protocoles (AH et ESP) donnent lieu à deux modes d'utilisation: le mode transport et le mode tunnel :

Dans le mode transport, les données associées à AH ou à ESP viennent se greffer sur le datagramme IP initial (c'est à dire celui qu'on aurait envoyé en l'absence d'IPsec). Le datagramme IP résultant contient un paquet AH ou bien ESP qui contient lui-même le contenu du paquet initial (un segment TCP par exemple). Il faut bien sûr que l'entête IP initial soit modifié pour que les équipements en charge du traitement soient à même de reconnaitre la « transformation IPsec ». Le champ numéro de protocole doit indiquer 50 ou 51 pour ESP ou AH en lieux et place par exemple de 6 (TCP) ou 17 (UDP) [RFC 790]. C'est l'en tête (AH ou ESP) qui indiquera le protocole encapsulé qui était auparavant indiqué dans l'en-tête IP:

Le mode tunnel quant à lui donne lieu à la génération d'un nouveau datagramme IP réalisant une encapsulation AH ou ESP (qui contient lui-même le paquet IP initial sans modification). Dans ce mode, il y a donc en définitive deux entêtes IP. L'entête externe sera effectivement utilisé pour le routage dès l'émission du paquet. Quant à l'entête interne (non chiffré en AH et chiffré en cas

d'utilisation d'ESP), il ne sera examiné que par le destinataire : il doit être ignoré par les équipements réseau situés entre l'émetteur et le destinataire. On réalise ainsi un « tunnel » à travers un réseau.

5.2.1 Exploitation de notre formalisme pour le protocole AH

Le protocole AH est conçu pour assurer l'intégrité et l'authentification des datagrammes IP sans chiffrement des données. Comme nous l'avons rappelé, le protocole AH assure une transformation du datagramme IP en rajoutant un champ AH (Authentication Header). Dans ce champ, l'intégrité du datagramme est vérifiée à partir de la valeur du champ « AD » (Authentication Data). Comme nous l'avons également rappelé, le protocole AH peut donner lieu à une utilisation en mode tunnel ou en mode transport.

5.2.1.1 Spécification avec notre formalisme de AH en mode transport

En mode transport, l'entête AH se place entre l'entête IP et l'entête du protocole transport (Figure 35) et authentifie tout le datagramme IP sauf les champs mutables (c'est à dire les champs DSCP, ECN, Flags, Offset, TTL et Header checksum)

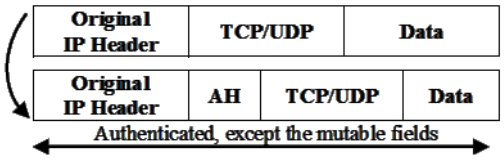

Figure 35. Datagramme IP avant et après AH en mode transport

En se basant sur le formalisme introduit dans les chapitres précédents, la transformation réalisée par le protocole AH de IPsec garantit l'intégrité du datagramme par adjonction du champ « ah ». En conséquence, cette transformation $tf_{AH}^{transport} : \mathcal{F} \rightarrow \mathcal{F}$, générant le flux est : $tf_{AH}^{transport} = f' = (< \cdots, \mathrm{ip}_1', \mathrm{ah}, \mathrm{p}, \ldots >, \mathrm{AUTHN}', \mathrm{CONF})$ où :

- $\mathrm{attributes}(\mathrm{ip}_1') = \mathrm{attributes}(\mathrm{ip}_1) \setminus \{(\mathrm{proto}, \mathrm{x})\} \cup \{(\mathrm{proto}, 51)\}$
  où 51 est la valeur du protocole AH.
- $\mathrm{AUTHN}' = \mathrm{AUTHN} \cup \cup_{\forall \mathrm{x} \in \mathrm{attributes}(\mathrm{ip}_1')\{\mathrm{DSCP,ECN,\ldots,checksum}\}} \{(\mathrm{ad}, ah, \mathrm{x}, \mathrm{ip}_1', \mathrm{s})\} \cup$
  $\cup_{\forall \mathrm{y} \in \mathrm{attributes}(ah)\{\mathrm{ad}\}} \{(\mathrm{ad}, ah, \mathrm{y}, ah, \mathrm{s})\} \cup$
  $\cup_{\forall \mathrm{p} \in \mathrm{rest}(ah) \mid \forall \mathrm{z} \in \mathrm{attributes(p)}} \{(\mathrm{ad}, ah, \mathrm{z}, \mathrm{p}, \mathrm{s})\}$

Ceci indique que l'intégrité de tous les champs non mutables du protocole IP, ainsi que tous les champs de tous les protocoles qui sont encapsulés par AH est garantie par l'utilisation de l'algorithme de sécurité « s ».

5.2.1.2 Spécification avec notre formalisme de AH en mode tunnel

En mode tunnel, un nouveau datagramme est généré. L'entête AH se place entre le nouvel entête IP et l'entête original IP. L'entête IP interne comporte les adresses ip source et destination ultimes, tandis que l'entête IP externe contient les adresses des homologues IPsec (Figure 36). Dans ce mode des garanties sont apportées sur le paquet IP initial y compris sur l'entête IP d'origine. En fait, en mode tunnel AH, l'entête IP d'origine et toute la donnée devient la charge utile « payload » pour le nouveau paquet. Le nouvel entête IP est protégé exactement de même que l'entête IP en mode de transport.

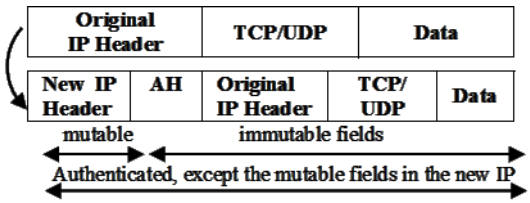

Figure 36. Datagramme IP avant et après AH en mode tunnel

L'utilisation de notre formalisme permet donc d'écrire cette transformation IPsec, comme suit :

$tf_{AH}^{tunnel}: \mathcal{F} \rightarrow \mathcal{F}$, génére le flux : $tf_{AH}^{tunnel}(f) = f' = (< \ldots, \mathrm{ip}_2, ah, \mathrm{ip}_1, \mathrm{p}, \ldots >, \mathrm{AUTHN}', \mathrm{CONF})$ où:

- $\mathrm{attributes}(\mathrm{ip}_2) = \{(\mathrm{ips}, \mathrm{sourcegateway}),\ (\mathrm{ipd}, \mathrm{destinationgateway}),\ (\mathrm{proto}, 51), \ldots\}$
- $\mathrm{AUTHN}' = \mathrm{AUTHN} \cup \cup_{\forall x \in \mathrm{attributes}(\mathrm{ip}_2)\setminus\{\mathrm{DSCP,ECN,\ldots,checksum}\}} \{(\mathrm{ad}, ah, \mathrm{x}, \mathrm{ip}_2, \mathrm{s})\} \cup$
  $\cup_{\forall y \in \mathrm{attributes}(ah)\setminus\{\mathrm{ad}\}} \{(\mathrm{ad}, \mathrm{ah}, \mathrm{y}, ah, \mathrm{s})\} \cup$
  $\cup_{\forall p \in \mathrm{rest}(ah) \mid \forall z \in \mathrm{attributes(p)}} \{(\mathrm{ad}, ah, \mathrm{z}, \mathrm{p}, \mathrm{s})\}$

Nous exprimons sous cette forme que l'intégrité de tous les champs non mutables du nouveau protocole IP ainsi que tous les champs de tous les protocoles qui sont encapsulés par AH (compris l'entête IP original) est garantie par l'utilisation de l'algorithme de sécurité « s ».

Donc, AH n'assure pas la confidentialité : les données sont signées mais pas chiffrées. Le paquet est authentifié par le checksum calculée à travers HMAC (Hash Message Authentication Code) [RFC 2104] en utilisant une clé secrète soit MD5-96 (Message-Digest) [RFC 2403], soit SHA-1-96 (Secure Hash algorithm) [RFC 2404].

5.2.1.3 Exemple de mise en œuvre d'un mode tunnel

A titre d'exemple la Figure 37 représente une situation où deux systèmes hôtes, chacun d'entre eux connecté à un réseau privé est autorisé à échanger avec l'autre à travers des firewalls mettant en œuvre le protocole IPsec. Une association de sécurité a été définie stipulant que tous les trafics TCP

entre « 10.0.0.0/8 et 20.0.0.0/8 » doivent être protégés par la mise en œuvre du protocole AH en mode tunnel avec l'algorithme cryptographique « HMAC-MD5 » et les paramètres cryptographiques nécessaires. La règle de politique écrite « *r* » est par exemple donnée sous la forme symbolique suivante :

r: « TCP 10.0.0.11 20.0.0.216 AH Tunnel 20.0.0.254 {hmac-md5} »

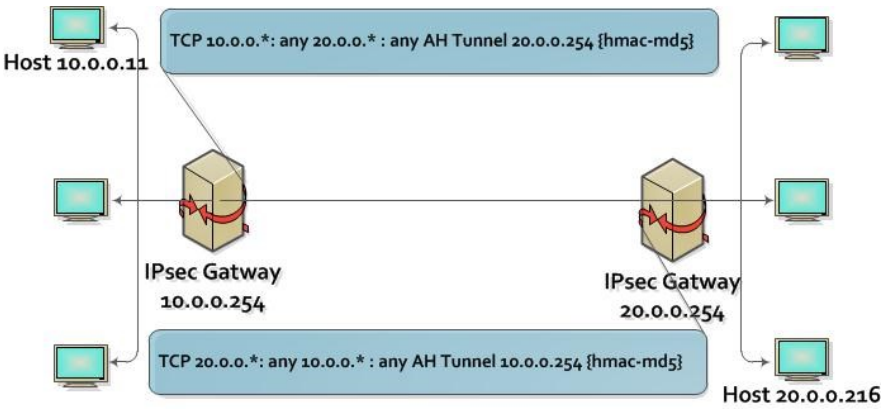

Figure 37. Présentation d'une règle IPsec

En utilisant notre formalisme (cf. chapitre 4 section 4.4), le mécanisme IPsec met ainsi en avant les deux composantes : *capacité* et *configuration*.

$$\text{IPsec} = < \text{CAPABILITY}_{\text{IPsec}}, \text{CONFIGURATION}_{\text{IPsec}} >$$

La *Capacité* décrit à travers le triplet : $< \boldsymbol{\Sigma_{IPsec}}, \mathbf{A_{IPsec}}, \mathbf{EXPR^{A}_{IPsec}} >$ le potentiel de IPsec (cf.0, Définition 3). Nous rappelons que :

- $\mathbf{A_{IPsec}} = \mathbf{A^{f}_{IPsec}} \cup \mathbf{A^{ctx}_{IPsec}}$ = {IP.proto, IP.ips, next(IP).ports, IP.ipd, next(IP).portd}

  Les attributs associés au protocole IPsec sont constitués d'éléments qui figurent dans le flux de données et d'autres qui figurent dans la règle :

  - $\mathbf{A^{f}_{IPsec}}$ représentant l'ensemble des attributs de flux de données $\boldsymbol{f}$ qui peut être trouvés par IPsec et,
  - $\mathbf{A^{ctx}_{IPsec}}$ est l'ensemble des attributs contextuels qui se trouvent dans toute règle de IPsec.
- $\boldsymbol{\Sigma_{IPsec}}$ = {IP_Address, PROTOCOL, STRING, BOOL, FLUX} répertorie l'ensemble des types non-vide et tel que chaque attribut figurant $\mathbf{A_{IPsec}}$ voit son type appartenir à $\boldsymbol{\Sigma_{IPSec}}$

La *configuration* est une liste des règles $\mathrm{RULE}^{\mathrm{A}}_{\mathrm{IPsec}} \subseteq \mathrm{CONDITION}^{\mathrm{A}}_{\mathrm{IPsec}} \times \mathrm{ACTION}_{\mathrm{IPsec}}$ (cf. chapitre 4,

Définition 5) et un algorithme de résolution des conflits « CRA ».

$r \in \mathrm{RULE}^{\mathrm{A}}_{\mathrm{IPsec}}$ est une règle de la configuration IPsec (figure 3) tel que $\mathbf{r = cond^{A}_{IPsec} \times action_{IPsec}}$ où :

- $[\mathrm{cond}^{\mathrm{A}}_{\mathrm{IPsec}}] = \{\mathbf{cond}_{\mathrm{IP.proto}} \wedge \mathbf{cond}_{\mathrm{IP.ips}} \wedge \mathbf{cond}_{\mathrm{next(IP).ports}} \wedge \mathbf{cond}_{\mathrm{IP.ipd.attributeValue}} \wedge \mathbf{cond}_{\mathrm{next(IP).portd}}\}$
- $\mathrm{action}^{\mathrm{A}}_{\mathrm{IPsec}}$= ApplyIPsec(*f*, AH, Tunnel, 20.0.0.254, hmac-md5)

Par la suite, la fonction ApplyIPsec sur la base de la valeur des paramètres fournis, doit appeler la fonction AH_Tunnel(dataflow, gateway, algorithm de chiffrement), puisque les paramètres « AH et Tunnel » figurent dans la liste des paramètres. Cette fonction appliquera parmi les 9 fonctions primitives que nous avons définies (cf section 4.3), celles qui conviennent afin d'obtenir le flux de sortie «*f'* »:

f' ← **AH_Tunnel** (*f* , gw, algo) {

Add_Proto(*f* , {AH_1,{nexthdr, … , ad}}, <IP,1>)

Add_Proto(*f*,{IP_2,{version,…,<proto,51>,…,ips,<ipd,end_tunnel>,option}},<AH,1>)

Add_AUTHN(*f,* ad, AH_1, x, IP_2, algo), ∀ x ∈ IP_2 \mutable

Add_AUTHN(*f,* ad, AH_1, y, AH_1, algo), ∀ y ∈ AH_1\{ad}

Add_AUTHN(*f,* ad, AH_1, z, IP_i, algo), ∀ z ∈ P and ∀ P ∈ rest(AH_1) et ∈ $\mathcal{P}$ | i ∈ $\mathbb{N}^*$

}

5.2.2 Exploitation de notre formalisme pour le protocole ESP

Le protocole ESP permet d'authentifier et de chiffrer les données transportées dans un datagramme IP. Tout comme le protocole AH, l'intégrité est vérifiée par rapport au champ « AD » et ESP peut donner lieu à une utilisation en mode transport ou en mode tunnel.

5.2.2.1 Spécification avec notre formalisme de ESP en mode transport

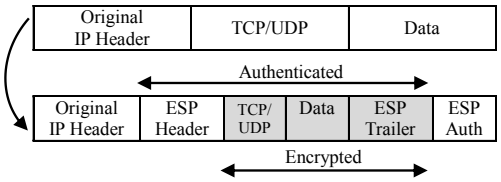

Figure 38. Datagramme IP avant et après ESP en mode transport

En mode transport, l'entête ESP encadre le protocole transport (Figure 38). ESP authentifie les données transportées dans le datagramme IP mais pas l'entête IP. De plus, il chiffre les données du protocole de la couche transport.

En se basant sur le formalisme précédent, nous considérons que les opérations menées au sein des protocoles donnent lieu à des transformations sous une forme globale, quand bien même les choix d'implémentation des protocoles conduisent parfois à ne pas positionner toutes les valeurs ajoutées au même endroit. Ainsi, dans la mise en œuvre de ESP nous considérons donc comme un tout la partie entête et la partie queue de ESP.

Ainsi, l'application du protocole ESP en mode transport sur un flux f est la fonction de transformation : $tf_{ESP}^{transport} : \mathcal{F} \rightarrow \mathcal{F}$, générant le flux $tf_{ESP}^{transport} = f' = (< \ldots, \mathrm{ip}_1', \mathrm{esp}, \mathrm{p}, \ldots >, \mathrm{AUTHN}', \mathrm{CONF}')$ où :

- $\mathrm{attributes}(\mathrm{ip}_1') = \mathrm{attributes}(\mathrm{ip}_1) \setminus \{(\mathrm{proto}, \mathrm{x})\} \cup \{(\mathrm{proto}, 50)\}$
  où 50 est la valeur pour le protocole ESP
- $\mathrm{AUTHN}' =$
  $\mathrm{AUTHN} \cup \bigcup_{\forall x \in \mathrm{attributes(esp)} \setminus \{ad\}} \{(\mathrm{ad}, \mathrm{esp}, \mathrm{x}, \mathrm{esp}, \mathrm{s}_1)\} \cup$
  $\bigcup_{\forall p \in \mathrm{rest(esp)} | \forall y \in \mathrm{attributes(p)}} \{(\mathrm{ad}, \mathrm{esp}, \mathrm{y}, \mathrm{p}, \mathrm{s}_1)\}$

  Ce qui indique que l'intégrité de tous les champs de tous les protocoles qui sont encapsulés par ESP est garantie par l'utilisation de l'algorithme de sécurité s_1, par exemple MD5.
- $\mathrm{CONF}' =$
  $\mathrm{CONF} \cup \bigcup_{\forall x \in \mathrm{attributes(esp)} \setminus \{spi,seq,ad\}} \{(\mathrm{x}, \mathrm{esp}, \mathrm{s}_2, \mathrm{all})\} \cup$
  $\bigcup_{\forall p \in \mathrm{rest(esp)} | \forall y \in \mathrm{attributes(p)}} \{(\mathrm{y}, \mathrm{p}, \mathrm{s}_2, \mathrm{all})\}$.

  Ceci indique que tous les champs de tous les protocoles qui sont encapsulés par ESP sont chiffrés par l'utilisation de l'algorithme de sécurité s_2, par exemple 3DES [RFC 1851].

5.2.2.2 Spécification avec notre formalisme de ESP en mode tunnel

En mode tunnel, l'entête IP interne porte l'adresse ultime ip source et destination, tandis qu'un entête IP externe contient les adresses des passerelles IPsec (Figure 39) et protège le paquet IP interne tout entier, incluant l'entête IP interne. La position de l'ESP en mode tunnel encadre le paquet IP original. L'intégrité du datagramme est vérifiée par rapport aux données d'authentification du champ (AD). En fait, en mode tunnel ESP, l'entête IP d'origine et toutes données deviennent la charge utile « payload » pour le nouveau paquet. Le nouvel entête IP n'est pas protégé comme l'entête IP en mode de transport.

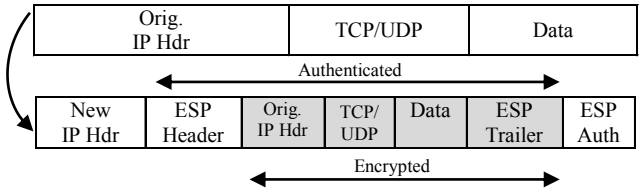

Figure **39. IP datagram before and after applying ESP in tunnel mode**

Ainsi, l'application du protocole ESP en mode tunnel sur un flux f est la fonction de transformation : $tf_{ESP}^{tunnel}: \mathcal{F} \rightarrow \mathcal{F}$, génére le flux $tf_{\mathrm{ESP}}^{\mathrm{tunnel}}(f) = f' = (< \ldots, \mathrm{ip}_2, \mathrm{esp}, \mathrm{ip}_1, \mathrm{p}, \ldots >, \mathrm{AUTHN}', \mathrm{CONF}')$ où :

- $\mathrm{attributes}(\mathrm{ip}_2) = \{(\mathrm{ips}, \mathrm{SourceGateway}), (\mathrm{ipd}, \mathrm{DestinationGateway}), (\mathrm{proto}, 50), \ldots\}$
- $\mathrm{AUTHN}' =$

  $\mathrm{AUTHN} \cup \cup_{\forall x \in \mathrm{attributes(esp)}\{\mathrm{ad}\}} \{(\mathrm{ad}, \mathrm{esp}, \mathrm{x}, \mathrm{esp}, \mathrm{s}_1)\} \cup$

  $\cup_{\forall p \in \mathrm{rest(esp)} \mid \forall y \in \mathrm{attributes(p)}} \{(\mathrm{ad}, \mathrm{esp}, \mathrm{y}, \mathrm{p}, \mathrm{s}_1)\}$

  Ce qui indique que l'intégrité de tous les champs de tous les protocoles qui sont encapsulés par ESP est garantie par l'utilisation de l'algorithme de sécurité s_1 à l'aide des fonctions de hachage MD5 ou SHA1.

- $\mathrm{CONF}' =$

  $\mathrm{CONF} \cup \cup_{\forall x \in \mathrm{attributes(esp)}\{\mathrm{spi,seq,ad}\}} \{(\mathrm{x}, \mathrm{esp}, \mathrm{s}_2, \mathrm{all})\} \cup$

  $\cup_{\forall p \in \mathrm{rest(esp)} \mid \forall y \in \mathrm{attributes(p)}} \{(\mathrm{y}, \mathrm{p}, \mathrm{s}_2, \mathrm{all})\}$

  Ceci indique que tous les champs de tous les protocoles qui sont encapsulés par ESP sont chiffrés par l'utilisation de l'algorithme de sécurité s_2 comme par exemple, 3DES-CBC (Triple DES en mode cipher block chaining) [RFC 2451] ou AES-CBC (Advanced Encryption Standard en mode cipher block chaining) [RFC 3394]et [RFC 3602].

5.3 Exploitation de notre formalisme pour la spécification du NA(P)T

Le NAT [RFC 3022] est principalement utilisé aujourd'hui (mais pas exclusivement) pour transformer l'adresse IP source d'un datagramme IP émis par un système inscrit dans un réseau privé, avec des équipements distant possédant des adresses IP privées ou non et le plus souvent accessibles en traversant le réseau Internet. En conséquence, ces adresses privées ne sont pas routables sur Internet et ne doivent pas être utilisées par des machines de ce réseau. La Figure 40 illustre l'un de ces mécanismes régulièrement mis en œuvre.

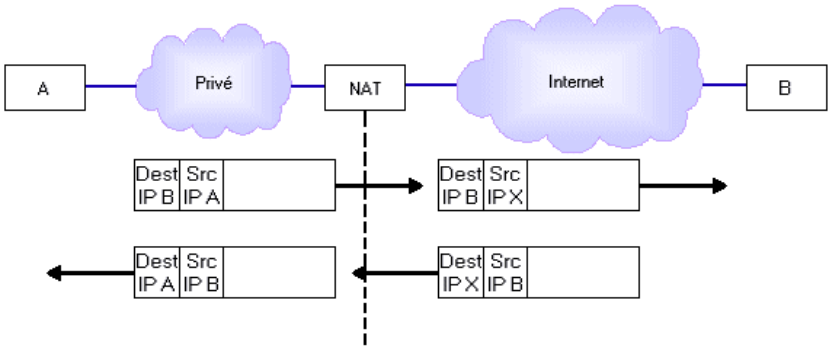

Figure 40. Fonctionnement de NAT [SecurityInfo]

Ici le NAT est effectuée en sortie d'un réseau pour gagner l'autre (du réseau privé au réseau Internet, ou du réseau Internet vers le réseau privé). Le NAT va effectuer le remplacement de l'IP source de A par son « IP X » puis il va router le paquet vers le réseau extérieur. Lorsqu'un datagramme arrivera à destination d'un système du réseau privé c'est l'opération inverse qui sera réalisée. Plusieurs formes de translation sont possibles (NAT de base, NAT dynamique, NAPT, Twice-NAT, etc).

Le NAT n'est pas une opération anodine et ce, bien que cette fonctionnalité ait pour vocation d'être transparente. En effet, le NAT modifie les paquets IP et cela a pour conséquence directe de remettre en cause tout contrôle d'intégrité au niveau IP et même aux niveaux supérieurs puisque TCP par exemple inclue les adresses dans ses checksum! Concrètement, on se rend compte qu'un protocole de sécurisation des datagrammes comme IPsec est totalement incompatible avec le NAT, que ce soit en mode tunneling ou en mode transport [SecurityInfo].

Nous avons choisi NAPT un de ces techniques les plus utilisées. Le fonctionnement de ce système peut être résumé ainsi lorsqu'un datagramme arrive depuis le réseau privé par :

- Le système NAPT génère un port source dynamiquement
- Le système NAPT enregistre l'association (ancienne adresse source, ancien port source, nouvelle adresse, nouveau port)

- Le système NAPT modifie les champs port source et checksum du protocole UDP/TCP
- Le système NAPT modifie l'adresse IP source et le checksum dans l'entête IP.

5.3.1 Spécification avec notre formalisme de NAPT Trivial

Par conséquent, nous pouvons représenter le traitement d'un système NAPT par la fonction de transformation tf^{S}_{NAPT} où :

- Pré-condition 1 : le protocole capable de traiter le contenu du datagramme IP est tcp, ou encore udp. Autrement dit dans la chaine d'encapsulation, le protocole suivant ip doit être un protocole de transport.

  $\forall f = (< \cdots \text{ip}, \text{succ(ip)} \ldots >, \text{AUTHN}, \text{CONF})$, avec :

  $\{\text{ports}, \text{portd}, \text{checksum}\} \in \text{attributes(succ(ip))}$

- Pré-condition 2 : le système NAPT doit être capable de lire les champs adresse IP source et port source :

  $\forall f = (< \cdots \text{ip}, \text{succ(ip)} \ldots >, \text{AUTHN}, \text{CONF}) =>$

  $\nexists \text{s} \mid (\text{ips}, \text{ip}, \text{s}, \text{all}) \in \text{CONF} \vee (\text{ports}, \text{succ(ip)}, \text{s}, \text{all}) \in \text{CONF}$

Le système NAPT transforme les champs adresse IP source, checksum du IP, port source et checksum du protocole transport. Par conséquent, tf_{NAPT} transforme un flux $f = (< \cdots \text{ip}, \text{succ(ip)} \ldots >, \text{AUTHN}, \text{CONF})$ en un flux de données $f' = (< \cdots \text{ip}', \text{succ(ip)}' \ldots > , \text{AUTHN}, \text{CONF})$ tel que:

- $\text{attributes}(\text{ip}') =$
  $\text{attributes(ip)} \setminus \{(\text{ips}, \text{value}), (\text{checksum}, \text{value})\}$
  $\cup \{(\text{ips}', \text{new\_value}), (\text{checksum}', \text{new\_value})\},$
- $\text{attributes}(\text{next}(\text{ip})') =$
  $\text{attributes}\big(\text{next(ip)}\big) \setminus \{(\text{ports}, \text{value}), (\text{checksum}, \text{value})\}$
  $\cup \{(\text{ports}', \text{new\_value}), (\text{checksum}', \text{new\_value})\}$

Dans ce que nous avons défini comme le NAPT trivial, le traitement considère uniquement que seuls le protocole TCP ou le protocole UDP suit immédiatement IP. L'accès aux numéros de port est donc possible. Cependant, l'évolution dans le temps de l'agencement des encapsulations de protocole ne permet pas de garantir que cette situation sera toujours avérée. Nous verrons plus loin que la conjugaison d'IPsec et de NAPT pose un problème lorsque l'on veut réaliser une opération de substitution de numéros de ports. Nous proposons donc une deuxième version plus évoluée qui n'est pas limitée par cette hypothèse disant que TCP ou UDP suit IP. Dans cette version évoluée, nous

considérons que le parcours des encapsulations doit permettre de rechercher l'entête TCP ou UDP plus loin dans le flux de données.

5.3.2 Spécification avec notre formalisme de NAPT évolué

La fonction de transformation tf^{e}_{NAPT} représentant un système NAPT évolué est défini ainsi :

- Pré-condition 1 : le protocole IP encapsule directement ***ou indirectement*** le protocole TCP ou UDP :

  $\forall f = (< \cdots \text{ip}, \ldots, \text{suite(ip)}, \ldots >, \text{AUTHN}, \text{CONF}), \exists \text{p} \in \text{suite(ip)} \mid$

  $\{\text{ports}, \text{portd}, \text{checksum}\} \in \text{attributes(p)}$

- Pré-condition 2 : le système NAPT doit être capable de lire les champs adresse IP source et port source :

  $\forall f = (< \cdots \text{ip}, \ldots, \text{suite(p)}, \ldots >, \text{AUTHN}, \text{CONF}), \nexists \text{s tel que } (\text{ips}, \text{ip}, \text{s}, \text{all}) \in \text{CONF} \vee$

  $(\text{ports}, \text{suite(ip)}, \text{s}, \text{all}) \in \text{CONF}$ où suite(ip) est un protocole de transport.

Le système NAPT transforme les champs adresse IP source et checksum du IP et port source et checksum du protocole transport. Par conséquent, tf_{NAPT} transforme un flux $f = (< \cdots, \text{ip}, \ldots \text{p} \ldots >, \text{AUTHN}, \text{CONF})$ en un flux de données $f' = (< \cdots \text{ip}', \ldots \text{p}' \ldots >, \text{AUTHN}, \text{CONF})$ tel que:

- $\text{attributes(ip}') =$
  $\text{attributes(ip)} \setminus \{(\text{ips}, \text{value}), (\text{checksum}, \text{value})\}$
  $\cup \{(\text{ips}', \text{new\_value}), (\text{checksum}', \text{new\_value})\},$
- $\text{attributes(p}') =$
  $\text{attributes(p}') \setminus \{(\text{ports}, \text{value}), (\text{checksum}, \text{value})\}$
  $\cup \{(\text{ports}', \text{new\_value}), (\text{checksum}', \text{new\_value})\}$

5.3.3 Exemple de mise en œuvre de NA(P)T

Supposons qu'au sein d'un firewall, l'administrateur souhaite protéger les adresses ip destination et les numéros de port destination utilisés en internet. Pour cela il écrit une règle « *r* » de la forme suivante :

r: « #iptables -t nat -A PREROUTING -p tcp -i eth1 -dport 80 -j NAT --to-destination 123.123.123.123:22 »

Cette règle « *r* » remplace ces champs adresse ip destination et port destination par la valeur d'adresse 123.123.123.123 et le numéro de port 22.

Suivant notre formalisme, cette opération de transformation portant sur les flux de données est notée : $tf_{\mathrm{NA(P)T}}(f)$, avec le flux de données $= (< \ldots, \mathrm{ip}_1, \mathrm{p}_1, \ldots >, \{\}, \{\})$. Cette transformation donnera le flux de données $f' = tf_{\mathrm{NA(P)T}}(f) = (< \ldots, \mathrm{ip}'_1, \mathrm{p}'_1, \ldots >, \{\}, \{\})$.

De la même façon que nous l'avons fait pour IPsec, nous allons illustrer la construction des composantes capacités et configuration en s'appuyant sur notre formalisme :

$\mathrm{NA(P)T} = < \mathrm{CAPABILITY}_{\mathrm{NA(P)T}}, \mathrm{CONFIGURATION}_{\mathrm{NA(P)T}} >$ (cf. chapitre 4, section 4.4.1),

- *Capacité* est défini par le triplet : $< \mathbf{\Sigma_{NA(P)T}}, \mathbf{A_{NA(P)T}}, \mathbf{EXPR^{A}_{NA(P)T}} >$ (cf. chapitre 4, Définition 3) qui donne les caractéristiques décrivant les capacités potentielles de NA(P)T comme suit :
  - $\mathbf{A_{NA(P)T}} = \mathbf{A^{f}_{NA(P)T}} \cup \mathbf{A^{ctx}_{NA(P)T}}$ = {preceding(IP).interface, IP.proto, IP.ipd, next(IP).portd, next(IP).checksum} tel que: $\mathbf{A^{f}_{NA(P)T}}$ est l'ensemble des attributs de flux de données ***f*** qui peut être récupéré par NA(P)T et représenté par $\mathbf{A^{ctx}_{NA(P)T}}$ qui est l'ensemble des attributs contextuels trouvé dans n'importe quelle règle de NA(P)T.
  - $\mathbf{\Sigma_{NA(P)T}}$ = {IP_Address, STRING, INTEGER, BOOL, FLUX} est l'ensemble des types non-vide tel que chaque attribut dans $\mathbf{A_{NA(P)T}}$ a un type qui appartient à $\mathbf{\Sigma_{NA(P)T}}$
- *Configuration* définit une liste des règles $\mathrm{RULE}^{\mathrm{A}}_{\mathrm{IPsec}} \subseteq \mathrm{CONDITION}^{\mathrm{A}}_{\mathrm{IPsec}} \times \mathrm{ACTION}_{\mathrm{IPsec}}$ (cf. chapitre 4, Définition 5) et un algorithme de résolution des conflits « CRA ».

$r \in \mathrm{RULE}^{\mathrm{A}}_{\mathrm{NA(P)T}}$ est une règle de la configuration NA(P)T tel que $\mathbf{r = cond^{A}_{NAPT} \times action_{NAPT}}$ où :

- $[\mathrm{cond}^{\mathrm{A}}_{\mathrm{NAPT}}]$= true et
- $\mathrm{action}^{\mathrm{A}}_{\mathrm{NA(P)T}}$ = ApplyNAPT(*f*, 123.123.123.123, 4567)=

  {

  Modify_Attribute(*f*, <@ipd, 123.123.123.123>, <IP,1>)

  Modify_Attribute(*f*, <portd, 22>, next(<IP,1>).protoName)

  Modify_Attribute(*f*,<checksum,new-checksum()>, next(<IP,1>).protoName)

  }

Si l'expression booléen $[\text{cond}^{A}_{\text{NAPT}}]$ est satisfaite et validée, alors $\text{action}_{\text{NAPT}}$ doit s'exécuter en appelant la fonction « Modify_Attribute() » par trois fois afin de modifier :

1) l'adresse destination « @ipd » du protocole IP_1 ,
2) le port destination « portd » du protocole du transport TCP_1 et
3) le checksum du protocole du transport TCP_1.

5.4 Rappels sur l'architecture iptables/netfilter

Un système de firewall fonctionne sur le principe du filtrage de paquets. Compte tenu de la large diffusion de la pile de protocole TCP/IP, nombre de firewall sont aujourd'hui constitués par des systèmes analysant les entêtes des datagrammes IP. Ces opérations d'analyse peuvent aussi bien porter sur des datagrammes générés au sein d'un réseau privatif, que de datagrammes venant de réseaux externes et en particulier de l'Internet. Dans l'environnement Linux, Netfilter [Netfilter] est le module logiciel qui donne la possibilité de contrôler, modifier et filtrer les paquets IP, et d'assurer un suivi de connexions établies entre systèmes. Ce module est présent au sein des noyaux Linux depuis la version 2.4. Iptables est l'interface « ligne de commande » permettant de configurer Netfilter. D'autres modules tels ip6_tables, arp_tables et ebtables existent également. Ce sont des systèmes d'accroches de Netfilter basés sur des tableaux qui contiennent des règles de firewall conduisant à un filtrage ou à une transformation des paquets reçus. Tous les paquets inspectés par Netfilter passent à travers des tables de traitement prédéfinies, qui sont des sortes de files d'attente de traitement des paquets. Chacune de ces files d'attente est dédiée à un type particulier d'activité et est contrôlé par une chaîne associée, qui a pour rôle de transformer ou de filtrer le paquet.

- La table « RAW » peut être utilisé pour filtrer des paquets avant qu'ils n'atteignent les opérations nécessitant plus de mémoire,
- La table « MANGLE » permet d'apporter des modifications au paquet, qui peuvent influencer d'autres règles, telles que le NAT ou le filtrage.
- La table « NAT » permet d'assurer les opérations de modification des adresses source et destination.
- La table « FILTER » permet d'assurer le filtrage des paquets

Au sein de chaque table, des chaînes sont définies. Une chaine est une liste de règles qui peuvent concerner un ensemble de paquets. Chaque règle spécifie ce qui doit être opéré sur un paquet. Les chaines « PREROUTING, INPUT, FORWARD, OUTPUT et POSTROUTING » sont prédéfinies. Des chaines définies par l'utilisateur peuvent être ajoutées. A titre d'exemple, la table NAT utilise les chaines prédéfinies :

- PREROUTING : Les paquets vont être modifiés à l'entrée de la pile réseaux, et ce, qu'ils soient à destination des processus locaux où d'une autre interface.
- OUTPUT : Les paquets sortant des processus locaux sont modifiés.
- POSTROUTING : les paquets qui sont prêts à être envoyés aux interfaces réseaux sont modifiés

Au sein d'un tel système iptables, les paquets sont amenés à être examinés par différentes chaines. Par exemple « *f* » dans la Figure 41 caractérise un paquet qui provient du réseau. Ce paquet traité dans un premier temps par la chaine « PREROUTING » donnera ou non-lieu à des modifications ou un traçage en fonction des informations contenues dans les tables RAW, MANGLE et NAT. Puis, en fonction du résultat de la décision de routage (routing decision), le paquet sera envoyé soit à la chaine « INPUT » lorsque la destination est l'hôte local (local host), soit aux chaînes « FORWARD » et « POSTROUTING » si la destination est un hôte distant (remote host). Si un processus local (local process) envoie un paquet, il passe à travers les chaînes « OUTPUT » et « POSTROUTING » [frozentux].

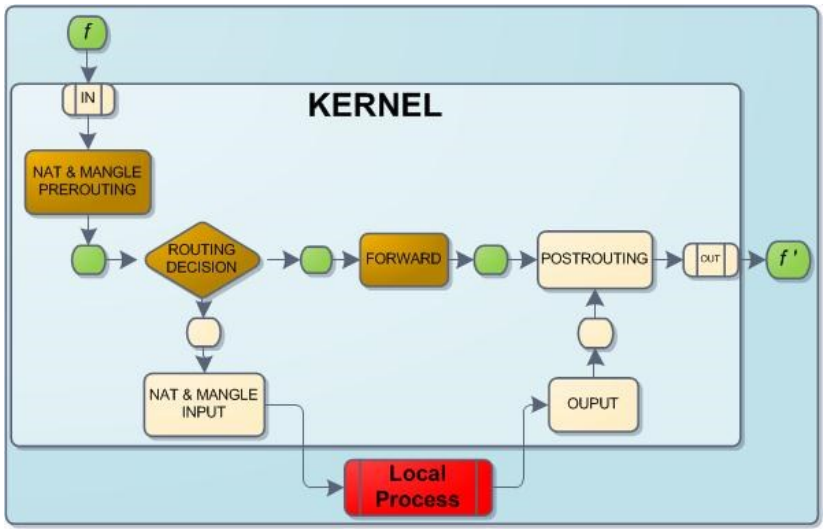

Figure 41. Noyau iptables

Bien que notre formalisme puisse être utilisé plus largement, nous limiterons son illustration à l'exploitation de la table MANGLE dans le cadre iptables.

Une règle « MANGLE » (le terme mangle est en anglais, veut dire mutilation en français) permet de mettre des marques sur les paquets en y attachant une information. Ces marques peuvent être utilisées pour un traitement ultérieur (future processing) qui peut utiliser cette marque dans les conditions de leurs règles. Ils identifient un paquet basé sur sa marque et la traite en conséquence. L'idée de cette technique est par exemple de fournir à Linux la possibilité d'avoir un contrôle sur les débits des flux de données entrants et sortants de la machine, afin de rendre certains flux plus

prioritaires que d'autres[7]. Les marques de « MANGLE » n'existent que dans le même système Linux impacté et ne peuvent être transmises à travers le réseau. En outre, la facilité de « MANGLE » est utilisée pour modifier certains champs dans l'en-tête IP, comme TOS (Type de Service (DSCP)) et le champ TTL (Time To Live).

Dans les nouvelles versions des noyaux, « MANGLE » utilisent toutes les chaines du filtre IP :

- PREROUTING : Les paquets vont être marqués en entrée de la couche réseau, en fonction de certains critères, de type de service (grâce aux numéros de ports source et/ou de destination), d'adresses IP de source et/ou de destination, de taille des paquets, etc. Ces informations seront utilisées par un programme fonctionnant dans l'espace noyau.
- INPUT : Les paquets sont marqués juste avant d'être envoyés aux processus locaux.
- FORWARD : Les paquets passant d'une interface réseau à l'autre sont marqués.
- OUTPUT : Là, ce sont les paquets générés par les applications locales (un client web par exemple) qui vont être marqués, tout comme les paquets entrant dans la couche réseau.
- POSTROUTING : Les paquets prêts à être envoyés sur le réseau sont marqués. L'utilisation de cette chaîne dans la table « MANGLE » n'est cependant pas très évidente.

5.5 Etudes de cas

A travers notre travail, nous souhaitons démontrer que nous sommes à même de pouvoir détecter les conflits qui peuvent naître lorsque des flux d'information donnent lieu à des opérations de transformations. Nous allons illustrer l'intérêt de ce travail en considérant des combinaisons où les fonctionnalités de IPsec, de NAPT et Netfilter/iptables sont conjuguées.

Bien que nous sachions par avance que la combinaison de IPsec et de NA(P)T pose problème, l'idée reste ici d'illustrer les possibilités de détection d'erreurs. Il en est de même dans le cadre de l'utilisation conjointe des tables MANGLE, FILTER et NAT au sein de la technologie iptables. Dans les sections précédentes nous avons rappelé les caractéristiques principales de ces protocoles et nous avons donné leur représentation en utilisant le formalisme introduit dans les premiers chapitres.

Au-delà des phases de spécification, nous avons utilisé les réseaux de Pétri colorés en complément de notre approche de modélisation des fonctionnalités. L'objectif est de simuler les impacts provoqués sur un flux par les fonctionnalités afin de déterminer les inconsistances qui peuvent découler de la mise en œuvre d'une fonctionnalité.

[7] La table mangle peut nous conduire à un problème du sectarisme au niveau réseau ☺

Les trois sections suivantes illustrent l'articulation de mécanismes IPsec et de fonctionnalités de translation d'adresse réseau et de ports à travers trois scenarii :

- Un premier scénario illustre la mise en œuvre de notre proposition GAM sur les tunnels,
- Un deuxième scenario présente l'utilisation de l'ensemble AUTHN
- Un troisième scénario présente l'exploitation qui peut être faite de l'ensemble CONF

Enfin une quatrième et dernière section illustrera les conflits qui peuvent être détectés dans l'utilisation des tables MANGLE et FILTER de la technologie Netfilter/iptables.

Le but étant d'illustrer l'intérêt de nos travaux, nous avons souhaité faciliter la compréhension des exemples. Bien que dans la réalité, le nombre de protocoles puisse être bien plus important, nous avons limité à trois les encapsulations consécutives donnant lieu à la production de flux. Nous insistons sur notre volonté de démontrer que les conflits naissant de la combinaison de différentes fonctionnalités peuvent être détectés au moyen de notre modèle GAM et non pas de démontrer que ce modèle est simplement applicable à la mise en œuvre de IPsec, de NAPT et de iptables.

Par la suite, nos scénarii utiliseront les protocoles AH, ESP, IP, TCP. Les attributs de ces protocoles sont les suivants :

- attributes(ip_i)={vers, hlength, tos, tlength, id, flags,offset, ttl, proto, checksum, ips, ipd, option}
- attributes(tcp_i)={ports, portd, seq, ack, hlength, reserved, tcpflags, win, option,checksum}
- attributes(ah_i)={nexthdr, payloadlength, reserved, spi, seq, ad}
- attributes(esp_i)={spi, seq, padlength, nextheader, ad}

5.5.1 Scenario 1 : Un flux AH en mode Tunnel et NA(P)T

5.5.1.1 L'authentification

Dans ce scenario, nous étudions l'interaction entre un mécanisme qui met en œuvre AH (Authentication Header) en mode tunnel et le mécanisme NA(P)T. Ce scenario présentera l'encapsulation (tunneling) et montrera la détection de conflit en utilisant GAM sans avoir de connaissance a priori dans le domaine.

Notre étude consiste donc à analyser la chaine de transformation $tf_{NAPT} \circ tf_{AH}^{tunnel}(f): \mathcal{F} \rightarrow \mathcal{F}$, qui affecte un flux de données $F = (< \cdots, ip_1, tcp_1, \ldots >, \{\}, \{\})$ où les ensembles AUTHN et CONF sont vides.

La Figure 42 représente à l'état initial le réseau de Pétri coloré qui doit traiter le flux d'information « F ». Ce flux « F » est modifié par l'exécution du protocole IPsec2 noté ainsi dans le

réseau de Pétri et associé à la mise en œuvre du protocole AH en mode tunnel. Comme nous l'avons indiqué dans notre formalisme, les opérations de transformation du flux peuvent être impactées par des attributs contextuels « A_M^{ctx} » figurant dans une règle de configuration du mécanisme « M ». Nous considérons que ces attributs contextuels pourraient être communs à plusieurs mécanismes. En conséquence ils figurent dans « Memory ».

Le résultat « F' » est un flux authentifié qui doit traverser NA(P)T et dont naturellement la sortie devrait être « F'' ».

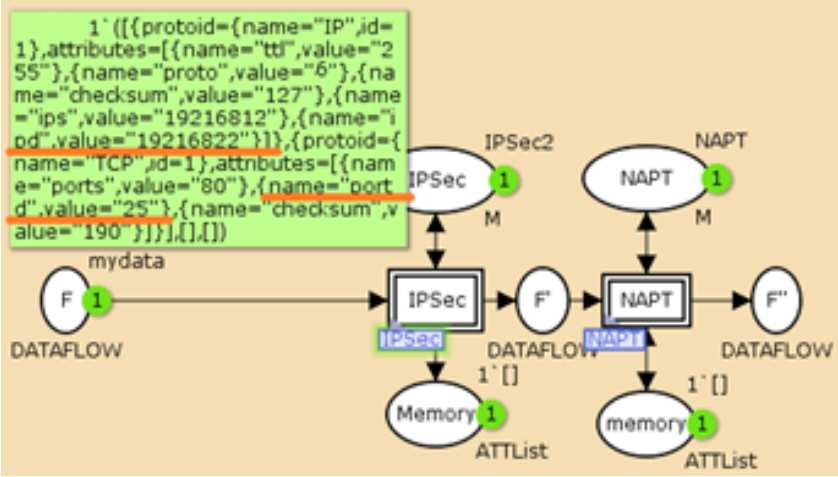

Figure 42. La détection des conflits : état initial

Nous supposons qu'après avoir injecté le flux « F » dans IPsec, la condition $[\text{cond}_{\text{AH}}^{\text{A}}]$ de l'application du protocole AH en mode tunnel est validée et doit donc être satisfaite.

En conséquence, l'$\mathbf{action}_{\mathbf{AH\_Tunnel}}$ doit alors être engagée. Elle nécessite l'exécution de la fonction ApplyIPsec (F, proto-IPsec, mode, gw, algo). Cette fonction est instanciée avec le flux de données « F », où « proto-IPsec » est le protocole AH, où le mode est tunnel, en donnant la passerelle impactée, et l'algorithme cryptographique à exécuter pour protéger ce qui doit l'être (cf section 5.2.1.3). L'exécution de cette fonction aura pour conséquence la création du flux « F' ». Nous rappelons ici que :

Le flux de données « F » a été transformé en « F' » (Figure 43) avec :

F' $= tf_{AH}^{tunnel}(\text{F}) = (< \ldots, ip_2, ah, ip_1', tcp', \ldots >, \text{AUTHN}, \{\})$ et où :

- $\text{attributes}(ip_2) = \{(ips, sourcegateway), (ipd, destinationgateway), (proto, 51), \ldots\}$
  ces valeurs sont soulignées en bleu et en haut dans la Figure 43.
- AUTHN $=$
  $\cup_{\forall x \in \text{attributes}(ip_2)\setminus\{ttl,tos,flags,\ldots\}}\{(ad, ah, x, ip_2, s)\} \cup$
  $\cup_{\forall y \in \text{attributes}(ah)\setminus\{ad\}}\{(ad, ah, y, ah, s)\} \cup \cup_{\forall z \in \text{attributes}(ip_1)}\{(ad, ah, z, ip_1, s)\} \cup$
  $\cup_{\forall t \in \text{attributes}(tcp)}\{(ad, ah, t, tcp, s)\}$

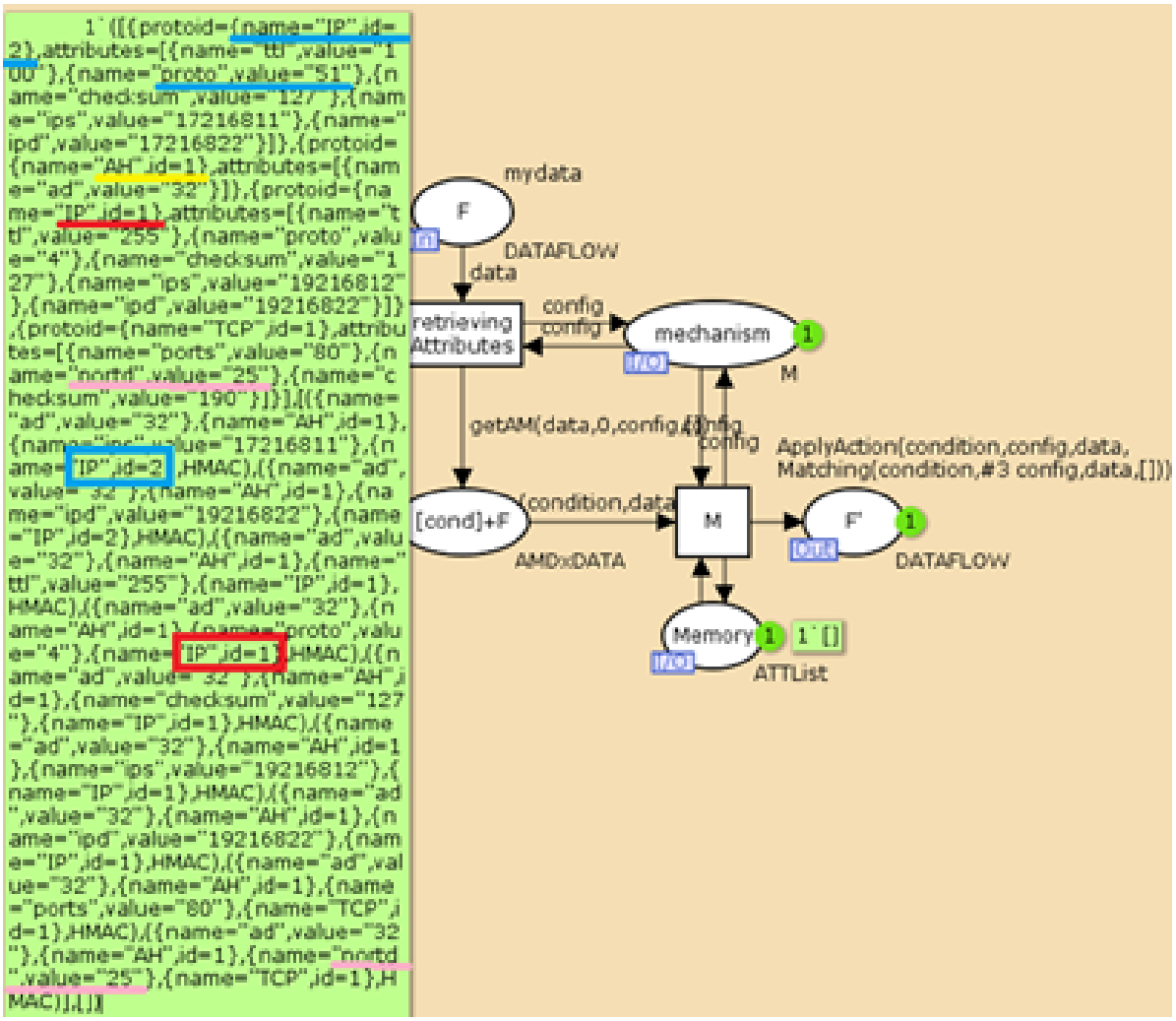

Figure 43. La détection des conflits : un flux de données après l'application de AH

5.5.1.2 L'analyse des conflits

A l'issue du passage par la fonctionnalité IPsec AH en mode tunnel, le flux « F' » est injecté dans le mécanisme NAPT. En conséquence la fonction $\mathbf{Action_{NAPT}}$, est instanciée. Trois appels consécutifs à la primitive de base Modify_Attribute() vont être effectués, l'un portant sur la demande de modification de l'adresse destination, le second pour modifier le port destination et enfin le dernier pour modifier l'attribut checksum (voir l'exemple de mise en œuvre d'un mode tunnel, cf. section 5.2.1.3).

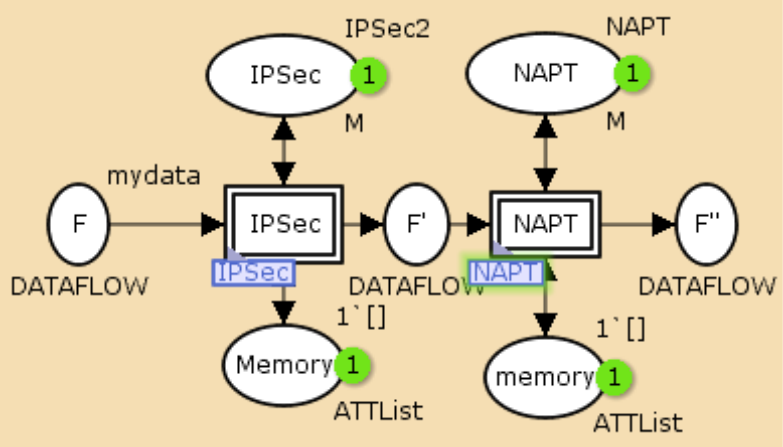

Figure 44. La détection des conflits : AH en mode tunnel bloqué dans NAPT

Lorsqu'on analyse le fonctionnement du réseau de Pétri coloré décrit dans la Figure 44, le jeton de couleur rouge associé au flux de données « F' » se trouve bloqué à l'intérieur du NAPT-GAM. En conséquence, les flux de données « F' » ne peut pas être transformé, que ce soit par la tentative de transformation par du NAPT trivial, ou bien par du NAPT évolué. Les raisons en sont les suivantes :

1) Le flux AH en passant par NAPT trivial :

La mise en œuvre du NAPT trivial n'est pas possible car la *pré-condition 1* n'est pas vérifiée. La pré-condition1 stipule que le protocole capable de traiter le contenu du datagramme est soit TCP, soit UDP. Si cette pré-condition est vérifiée, il est alors possible de mener une opération de substitution des numéros de ports par les appels consécutifs des primitives GetAttribute() et « Modify_Attribute() ».

Or le protocole cité comme celui capable de traiter le contenu du datagramme IP étant AH (valeur soulignée en jaune dans le flux « F' » de la Figure 43), AH est donc le protocole invoqué après IP_1 (matérialisé par le soulignement en rouge du flux « F' » dans la Figure 43), AH ne contenant aucun attribut nommé *ports* l'opération GetAttribute() ne retourne donc pas les attributs attendus et donc le changement de place n'est pas opéré. Il y a donc blocage ; le flux résultant « F'' » est vide.

2) Le flux AH en passant par NAPT évolué :

En supposant maintenant que le flux « F' » soit traité par une fonctionnalité NAPT évoluée, nous devrions être capables de récupérer la valeur des attributs *ports*. Les opérations de substitution devraient être réalisées et le blocage constaté dans la situation précédente devrait être levé. En conséquence l'obtention du flux de données « F'' » devrait être l'aboutissement du réseau de Pétri coloré. Nous rappelons que:

$F'' = tf_{NAPT}^{adv} \circ tf_{AH}^{tunnel}(F') = (< ip_2', ah, ip_1, tcp' >, AUTHN, \{\})$, où :

- $attributes(ip_2') = attributes(ip_2) \setminus \{(ips, old), (checksum, old)\} \cup \{(ips, new), (checksum, new)\}$
- $attributes(tcp') = attributes(tcp) \setminus \{(ports, old), (checksum, old)\} \cup \{(ports, new), (checksum, new)\}$

Dans notre formalisme de représentation d'un flux, nous avons introduit l'ensemble AUTHN, qui contient la liste des attributs pour lesquels les valeurs sont certifiées authentique. Rappelons que dans notre notation si (a_1, p_1, a_2, p_2, s) ∈ AUTHN, alors l'attribut a_1, du protocole p_1, garantit l'intégrité de l'attribut a_2, du protocole p_2 à travers l'algorithme de sécurité « s » (cf.Définition 2).

A l'issue du traitement assuré par le protocole AH de IPsec, nous avons les éléments suivant :

- $(ad, ah, (ips\_old), ip_1, s)$ ∈ AUTHN qui stipule que l'attribut *ad* du protocole *ah* protège le champ *ips* de valeur *old* généré par le protocole ip_1 en utilisant l'algorithme *s*
- $(ad, ah, (checksum\_old), ip_1, s)$ ∈ AUTHN qui stipule que l'attribut *ad* du protocole *ah* protège le champ *checksum* de valeur *old* généré par le protocole ip_1 en utilisant *s*
- $(ad, ah, (ports\_old), tcp, s)$ ∈ AUTHN qui stipule que l'attribut *ad* du protocole *ah* protège le champ *ports* de valeur *old* généré par le protocole *tcp* en utilisant l'algorithme *s*
- $(ad, ah, (checksum\_old), tcp, s)$ ∈ AUTHN qui stipule que l'attribut *ad* du protocole *ah* protège le champ *checksum* de valeur *old* généré par le protocole *tcp* en utilisant l'algorithme *s*

A l'issue de l'opération de transformation comme AH est utilisé en mode tunnel, un nouvel entête IP est calculé el le calcul du champ d'authentification ad est réalisé en tenant compte des valeurs contenues dans ce nouvel entête. En conséquence :

- $(ad, ah, (ips'\_new), ip_2, s)$ ∈ AUTHN qui stipule que l'attribut *ad* du protocole *ah* protège le champ *ips'* de valeur *new* généré par le protocole ip_2 en utilisant l'algorithme *s*
- $(ad, ah, (checksum'\_new), ip_2, s)$ ∈ AUTHN qui stipule que l'attribut *ad* du protocole *ah* protège le champ *checksum'* de valeur *new* généré par le protocole ip_2 en utilisant *s*

L'opération de transformation NAPT doit apporter une modification sur les valeurs *d'adresse et de numéro de port.* A l'issue de l'opération de transformation, comme nous l'avons rappelé plus haut

$F'' = tf_{NAPT}^{adv} \circ tf_{AH}^{tunnel}(F') = (< ip_2', ah, ip_1, tcp' >, AUTHN, \{\})$, où :

Comme nous avons considéré que la version de NAPT est évoluée, les pré-conditions sont satisfaites. L'encapsulation porte sur des datagrammes générés par TCP ou UDP, et l'accès aux champs d'adresses est théoriquement possible. Le problème d'intégrité va toutefois rendre impossible la substitution des adresse et numéro de port, car lors de l'invocation de la fonction Modify_Attribute(), les appels consécutifs à Get_Attribute() et Get_Protocol() vont détecter que ces attributs font partie de l'ensemble AUTHN, que ces attributs sont protégés par le champ ad du protocole ah exécuté en mode tunnel. A titre illustratif l'attribut « ports » appartient à l'ensemble AUTHN. Il figure souligner en couleur rose dans la Figure 43.

En conséquence l'exploitation de la transformation NAPT évoluée n'est pas possible, après exécution de AH en mode tunnel.

5.5.2 Scénario 2 : Un flux AH en mode transport

Dans ce scénario, nous présentons un mécanisme implémentant AH en mode transport (Figure 45) illustrant l'utilisation de l'authentification « AUTHN ». Nous considérons qu'un flux a été initié au cours de la mise en œuvre d'une application, qui successivement donne lieu à l'utilisation du protocole TCP, puis du protocole IP et qui se solde par la transmission d'une trame Ethernet. Nous souhaitons porter un regard critique sur les possibilités d'authentification du flux de données $f = (< \cdots, ip_1, tcp, \ldots >, \{\}, \{\})$ en passant par la chaine de transformation : $tf_{\text{NAT}}\ o\ tf_{\text{AH}}^{\text{transport}}\ (f)$.

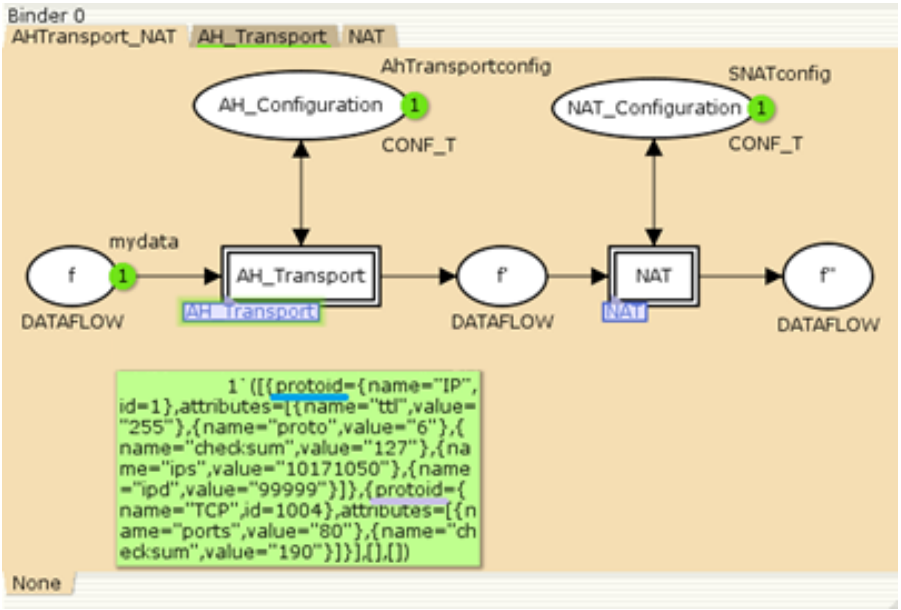

Figure 45. Détection de conflit : avant le passage par AH en mode transport

Le réseau de Pétri coloré présenté dans la Figure 45, prend en considération en entrée le flux *f*. Une première opération de transformation est réalisée par application du protocole de transport AH en mode transport. Au cours de cette opération de transformation du flux de données, le datagramme

ip_1 donne lieu à l'introduction de champs associés au protocole ah, puis à la fabrication d'un nouveau datagramme noté « ip_1' ». La définition de la condition dans ML (Standard Programming Language [Harper 2011]) doit être satisfaite pour que l'action soit déclenchée (cf. 0, Définition 6).

D'après la spécification de AH en mode transport (cf section 5.2.1.1), le flux de données f en passant par le protocole AH est en mode transport devrait être transformé en :

$f' = tf_{\text{AH}}^{\text{transport}}(f) = (< \cdots, ip_1', ah, tcp, \ldots >, \text{AUTHN}, \{\})$ (Figure 46) où dans un premier temps :

Il faut que l'entête IP initial soit modifié pour que les équipements en charge du traitement soient à même de reconnaitre la « transformation IPsec ». Si une communication utilisait des échanges TCP avec un numéro de port ayant pour valeur « 6 », le champ numéro de protocole doit maintenant indiquer « 51 » car associé au protocole AH, en lieu et place du numéro de protocole qui figurait précédemment (numéro 6).

- $\text{attributes}(ip_1') = \text{attributes}(ip_1) \backslash \{(proto, 6)\} \cup \{(proto, 51)\}$
  Par ailleurs, au cours de cette opération de transformation l'ensemble AUTHN vient indiquer qu'une protection en chaine des attributs de la liste des protocoles est assurée en exploitant assurée :
- $\text{AUTHN} = \cup_{\forall x \in \text{attributes}(ip_1')\{DSCP,ECN,\ldots,checksum\}} \{(ad, ah, x, ip_1', s)\} \cup$
  $\cup_{\forall y \in \text{attributes}(ah)\backslash\{ad\}} \{(ad, ah, y, ah, s)\} \cup \cup_{\forall z \in \text{attributes}(tcp)} \{(ad, ah, z, tcp, s)\}$

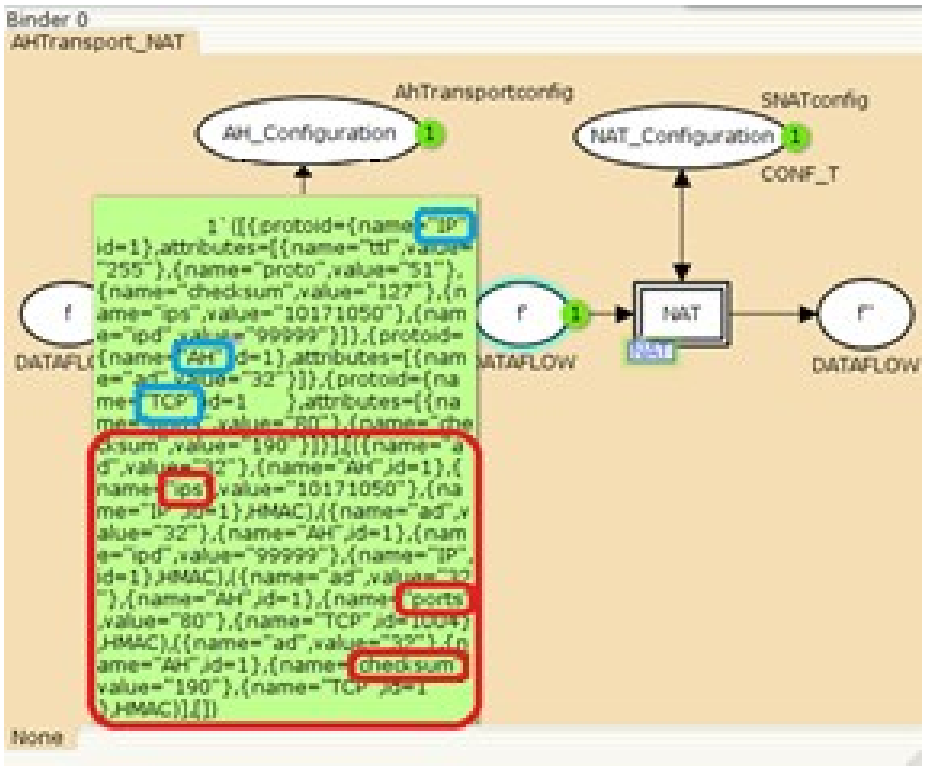

Figure 46. Détection de conflit : après le passage AH en mode transport

Le flux f' est obtenu à l'issue de la fonction de transformation AH_Transport; le protocole directement encapsulé après IP est AH encadré par le contour bleu dans la Figure 46. En conséquence, le jeton est associé au flux de données f' avec l'ensemble AUTHN (contenant les attributs authentifiés par l'attribut « ad » du protocole AH déjà encapsulé entre « ip_1' » et le protocole de transport « tcp ».

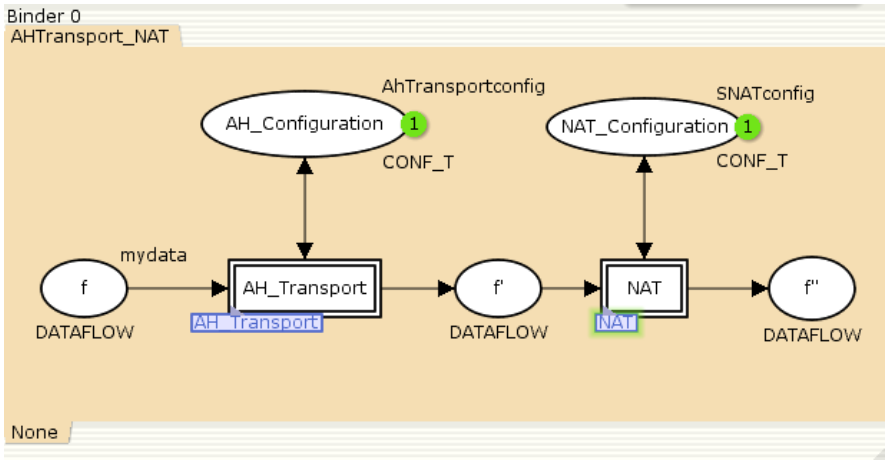

Figure 47. Détection de conflit : Conflit en passant par NAT

A l'issue du passage par la première fonction de transformation, le flux données f' est transmis à la deuxième fonction de transformation. Il devrait donner lieu à l'obtention du flux de données f''. D'après la Figure 47, le contour vert associé au petit encadré NAT indique que le jeton de flux de données f' a été bloqué. En effet ce flux de données f' ne peut pas être transformé, que ce soit par une configuration de la fonctionnalité NAT en mode « NAT trivial » (cf section 5.3.1) ou encore en « NAT évolué » (cf section 5.3.2) pour les raisons suivantes:

a) Avec AH à travers NAT trivial :

- Le flux de données f' ne respecte pas la pré-condition 1 de la spécification de NAT trivial ; i.e., Le protocole encapsulé directement après IP est AH qui n'est pas un protocole de transport.

b) Avec AH à travers NAT évolué, deux analyses sont possibles :

- Si f' se transforme par NAT, le resultat f'' n'est pas intègre (cf. Définition 2
- car la valeur des attributs ports et checksum diffère de celle qui appartient à l'ensemble d'authentification AUTHN
- Revenant au scénario, la 3ième exécution de la commande primitive (cf. Chapitre 40, section 4.3) : « Modify_Attribute() » ne peut pas exécuter car la pré-condition 2 (qui stipule que le système NAT doit être capable de lire les champs adresse IP source et port source) n'est pas respectée. En effet, l'attribut ports (respectivement l'attribut *checksum*) appartient à l'ensemble AUTHN (comme cela figure le grand contour

encadré en rouge dans la Figure 46). Dans cette situation, même si l'attribut ports peut être récupéré par la fonction « Modify_Attribute() », cet attribut ne peut pas être modifié pour se prémunir des problèmes d'intégrité. Le même raisonnement peut être appliqué pour l'attribut *checksum*.

5.5.3 Scénario 3 : un flux ESP en mode transport

Dans ce troisième scénario, nous présentons un mécanisme où sont appliqués tour à tour IPsec en mode ESP et le mécanisme NAPT (Figure 48). Nous modélisons en conséquence le fonctionnement d'un équipement qui reçoit un flux de données $f = (< \cdots, \mathrm{ip}_1, \mathrm{tcp}, \ldots >, \{\}, \{\})$ qui devrait être transformé par la chaine : $tf_{\mathrm{NAPT}}\ o\ tf_{\mathrm{ESP}}^{\mathrm{transport}}\ (f)$.

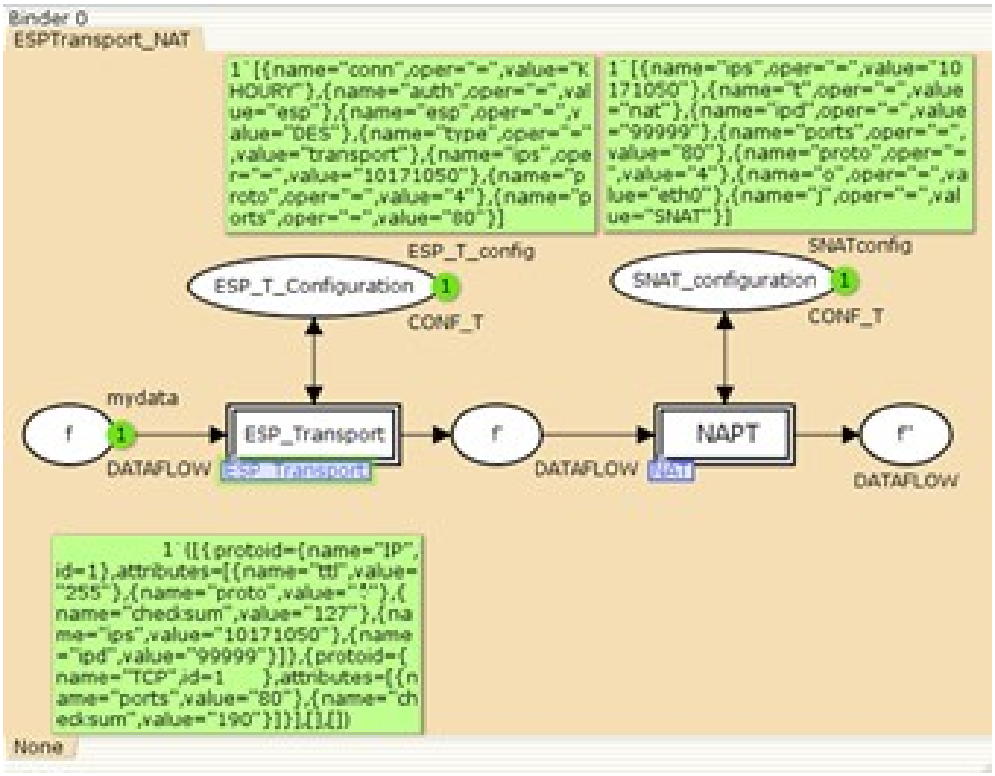

Figure 48. **Détection de conflit : avant le passage par ESP en mode transport**

D'après la spécification déjà présentée dans la section 5.2.2, le flux de données $f = (< \cdots, \mathrm{ip}_1, \mathrm{tcp}, \ldots >, \{\}, \{\})$ est transformé en $f' = \mathrm{tf}_{\mathrm{ESP}}^{\mathrm{transport}}(f) = (< \cdots, \mathrm{ip}_1', \mathrm{esp}, \mathrm{tcp}, \ldots > , \mathrm{AUTHN}, \mathrm{CONF})$ où :

Il faut que l'entête IP initial soit modifié pour que les équipements en charge du traitement soient à même de reconnaitre la « transformation IPsec ». Si une communication utilisait des échanges TCP avec un numéro de port ayant pour valeur « 6, » le champ numéro de protocole doit maintenant indiquer « 50 » car associé au protocole ESP, en lieu et place du numéro de protocole qui figurait précédemment (numéro 6).

- $\text{attributes}(\text{ip}_1') = \text{attributes}(\text{ip}_1) \setminus \{(\text{proto}, 6)\} \cup \{(\text{proto}, 50)\}$
- $\text{AUTHN} = \bigcup_{\forall x \in \text{attributes(esp)}\setminus\{\text{ad}\}} \{(\text{ad}, \text{esp}, x, \text{esp}, s_1)\} \cup \bigcup_{\forall y \in \text{attributes(tcp)}} \{(\text{ad}, \text{esp}, y, \text{tcp}, s_1)\}$,
- $\text{CONF} = \bigcup_{\forall x \in \text{attributes(esp)}\setminus\{\text{spi,seq,ad}\}} \{(x, \text{esp}, s_2, \text{all})\} \cup \bigcup_{\forall y \in \text{attributes(tcp)}} \{(y, \text{tcp}, s_2, \text{all})\}$.

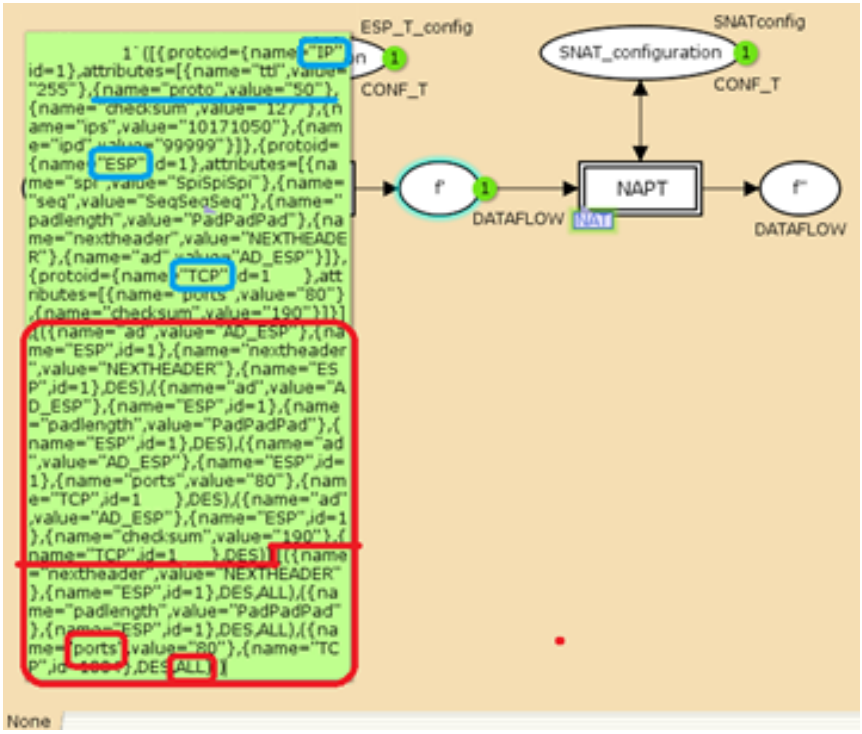

Figure 49. Détection de conflit : après le passage par ESP en mode transport

En utilisant la fonction de transformation NAPT trivial (cf section 5.3.1), il y a un conflit car f' ne vérifie pas la pré-condition 1 (qui stipule que le système NAPT doit être capable de traiter le contenu du datagramme IP et tcp, ou encore udp). En effet, le protocole directement encapsulé dans IP est ESP. En conséquence, le jeton représentant le flux de données est **bloqué**.

D'autre part, en utilisant le NAPT évolué (cf section 5.3.2*)*, la pré-condition 2 (qui stipule que le système NAPT doit être capable de lire les champs adresse IP source et port source) n'est pas satisfaite. En effet, au cours des opérations de transformation, l'ensemble CONF a été modifié et il est tel que : $(ports, tcp, s_2, all) \in \text{CONF}$ (encadré par le contour rouge en bas dans la Figure 49). Rappelons que l'appartenance à cet ensemble signifie que le port source ***ports*** du protocole ***tcp*** est complètement chiffré par application de l'algorithme s_2. Cet attribut n'est donc pas accessible le jeton reste bloqué dans NAPT (Figure 50).

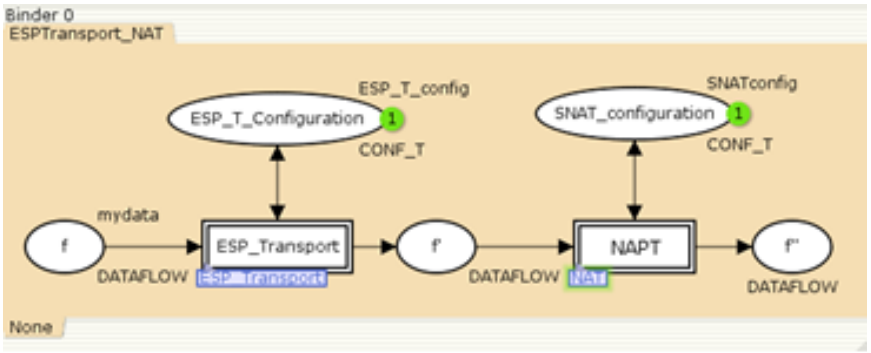

Figure 50. **Détection de conflit : avant le passage par NAPT**

5.5.4 Scénario 4 : Un flux mangle dans iptables

Supposons l'architecture d'un réseau d'entreprise (Figure 51), comprenant trois sous réseaux IP d'adresses 192.168.1.0, 192.168.2.0 et 192.168.3.0, trois routeurs identifiés R_1, R_2, R_3 et deux connexions différentes au réseau Internet accessibles derrière les routeurs R_2 et R_3.

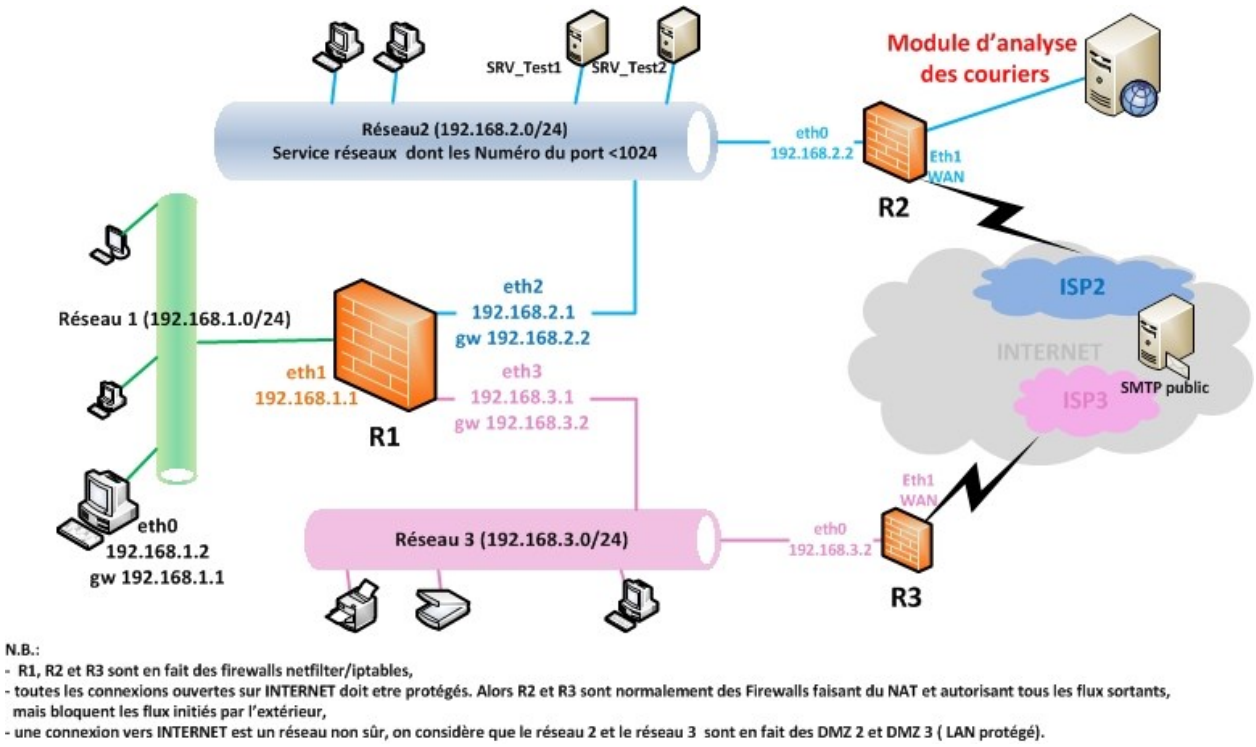

Figure 51. Scénario de configuration d'iptables

Deux modifications de politique viennent affecter dans le temps ce réseau.

1) ***Première modification de politique***

Dans un premier temps on impose à l'administrateur de capturer les trafics SMTP du réseau 192.168.1.0 pour le diriger vers R_2 car derrière ce routeur se trouve un module d'analyse des courriers (spam, virus, mots clés, signatures etc….).

L'administrateur décide d'installer sur R_1 Netfilter/iptables. Il choisit l'écriture de règles au niveau de l'équipement R_1 en conformité avec les possibilités qu'offre l'architecture Netfilter/iptables. Son choix se porte sur l'exploitation de la table MANGLE, un marquage conditionnel des paquets.

Afin de diriger le trafic SMTP vers le routeur R_2 d'adresse 192.168.2.2, l'administrateur exploite au sein du routeur R_1 la table MANGLE et la chaine PREROUTING qui donne la possibilité d'identifier par une marque le trafic SMTP. La marque positionnée doit conduire à modifier le routage de ce trafic. En conséquence l'administrateur crée la règle suivante (tableau 7) dans la chaine PREROUTING :

tableau 7. La règle iptables du marquage du trafic SMTP

```
# iptables -t mangle -A PREROUTING -p tcp --dport 25 -j MARK --set-xmark 0x0001/0xFFFF
```

Cette commande permet d'associer à un paquet (TCP et de numéro de port 25) une marque dont le bit b_0 est à « 1 » et où tous les autres bits de la marque sont à « 0 ». [Netfilter].

En complément l'administrateur doit organiser la redirection vers le réseau 192.168.2.0 des datagrammes qui sont impactés par ce marquage. Pour cela, conformément à la politique de routage en environnement Linux [Scottlowe] il faut que soit :

a) créée une table de routage spécifique à une stratégie (politique de routage)
b) créée une ou plusieurs règles indiquant au système laquelle des tables ce dernier doit exploiter pour déterminer la route qui doit être suivie,
c) Remplie la table de routage spécifique avec les valeurs qui conviennent.

tableau 8. La création de la table de routage

```
# echo 201 mail.out >> /etc/iproute2/rt_tables
# ip rule add fwmark 0x0001 table mail.out
# /sbin/ip route add default via 192.168.2.2 dev eth2 table mail.out
```

L'exécution des commandes suivantes (tableau 8) permet tout à tour de créer la table de routage « mail.out », d'indiquer au système que pour les datagrammes qui possèdent un marquage de valeur « 1 », les informations de routage se trouvent dans la table « mail.out », la dernière commande ajoutant la route par défaut vers 192.168.2.2 à travers l'interface eth_2.

2) ***Deuxième modification de politique***

Un nouveau changement de politique impose de protéger des serveurs de tests, installés sur le réseau 2 (192.168.2.0) et pour lesquels les numéros de ports TCP sont tous inférieurs à 1024. Seuls les utilisateurs des systèmes qui sont situés sur le réseau 2 (192.168.2.0) doivent avoir accès à ces systèmes de test. L'administrateur doit donc écrire une règle qui filtre ou non les paquets issus des réseaux 1 et 3 (192.168.1.0 et 192.168.3.0) dont l'adresse destination cible le réseau 2 (192.168.2.0) et dont le numéro de port destination est inférieur à 1024 soient supprimés ou rejetés.

Par ailleurs l'administrateur souhaite aussi que tout trafic destiné au réseau 2 en provenance des réseaux 1 et 3 donne lieu à une inscription dans le fichier de log.

L'administrateur décide de satisfaire à ces exigences en mettant en place de nouvelles règles Netfilter/iptables. Plusieurs solutions pourraient être adoptées. L'administrateur choisit là encore la technique du marquage des paquets et l'utilisation de la table MANGLE de R_1 et la chaine FORWARD. Tout paquet entrant qui possède une adresse destinataire 192.168.2.0/24 est marqué « 1 » (il marque le bit b_0), tout paquet dont le numéro de port destinataire (dport) est inférieur à 1024 est marqué « 2 » (il marque le bit b_1). Ainsi le bit « b_0 » de la marque caractérise l'appartenance au réseau 2 (192.168.2.0/24) et le bit « b_1 » de la marque caractérise un numéro de port inférieur à 1024.

L'administrateur exploite la valeur de la marque pour notifier dans le fichier de LOG et de manière discriminante les tentatives (succès ou échecs) de transmission des paquets depuis le réseau 1 (192.168.1.0) et depuis le réseau 3 (192.168.3.0).

L'élimination des paquets est alors prononcée pour ceux qui ont un numéro de port inférieur à 1024 (la valeur de la marque étant égale à « 3 » (b_1 b_0=11).

L'administrateur écrit l'enchaînement suivant de commandes iptables (tableau 9) qui réalise ce traitement :

tableau 9. Les règles iptables de 2ième modification

```
# iptables –t mangle -A FORWARD -d 192.168.2.0/24 -j MARK --set-xmark 0x0001/0x0000
# iptables –t mangle -A FORWARD -p tcp --dport 0:1024 -j MARK -- set-xmark 0x0002/0x0000
# iptables –t mangle -A FORWARD -s 192.168.1.0/24 -m mark -mark 0x0001/0x0001 -j LOG --log-
  prefix «  Dest_reseau_2 from reseau 1»
# iptables –t mangle -A FORWARD -s 192.168.3.0/24 -m mark -mark 0x0001/0x0001 -j LOG --log-
  prefix «  Dest_reseau_2 from reseau 3»
# iptables -A FORWARD -m mark -mark 0x0003  -j DROP
```

Si l'on considère la globalité de cette situation et de ces deux changements de politiques consécutifs dans le temps (gestion du trafic SMTP et filtrage ou non sur le réseau 2), on en vient naturellement à comprendre qu'il y a une incompatibilité dans les règles positionnées. Le trafic SMTP ne devrait pas passer sur le réseau 2 (192.168.2.0) puisque le numéro de port est inférieur à 1024.

5.5.4.1 Test de (non) conformité des exigences avec GAM-CPN

L'enchainement des commandes iptables est un cas de figure qui peut être appréhendé comme un flux de données transmis d'un élément à un autre. Les tableaux MANGLE, ROUTER et FILTER sont les éléments qui reçoivent des flux en entrée et fournissent des flux en sortie. Nous pouvons donc représenter la fonctionnalité iptables comme la chaîne de transformation suivante:

$$tf_{\text{iptables}}(f) = tf_{\text{MANGLE}} \circ tf_{\text{ROUTER}} \circ tf_{\text{MANGLE}} \circ tf_{\text{FILTER}}(f)$$

Par ailleurs chaque mécanisme de la technologie iptables peut donner lieu à la construction d'un réseau de Pétri coloré (CPN). En exploitant notre formalisme, chaque table de la chaîne d'iptables est modélisée à l'aide d'un GAM présentant des capacités et configurations (cf section 4.4). Des instanciations particulières des GAM sont associées aux tableaux : « MANGLE_1», « ROUTER », « MANGLE_2 » et « FILTER » (Figure 52).

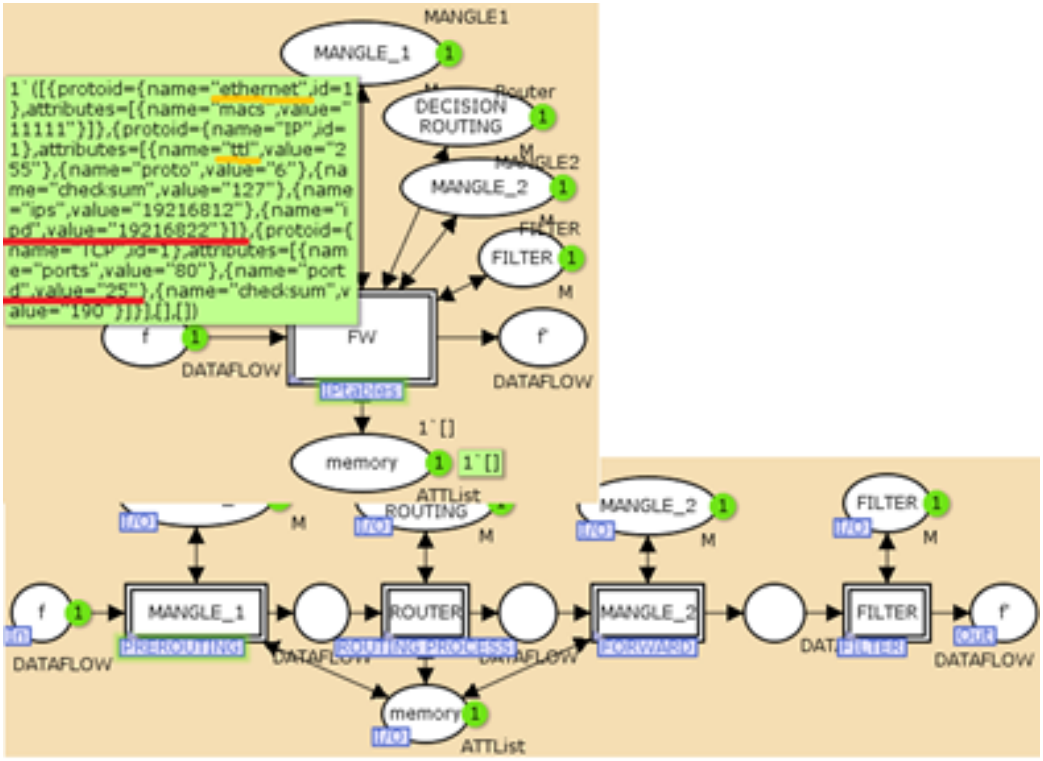

Figure 52. Navigation dans les menus de marquage

Différentes solutions pourraient être utilisées pour spécifier la valeur de la marque définie dans « MANGLE » et placée dans un paquet. Nous avons décidé de considérer que le positionnement de la marque dans le flux de données est le résultat de l'exécution d'un pseudo-protocole auquel nous avons

donné le nom de « mangle-mark » (meta-data) avec l'attribut « fwmark ». En approchant l'étude sous cette forme nous pouvons donc considérer que le premier tableau MANGLE_1 transforme un flux de données f =(<eth_1,ip_1,tcp_1>,{},{}) en un autre flux f' =(<(mangle-mark,{<*fwmark*,1>}),eth_1,ip_1,tcp_1>, {},{}).

Une autre approche aurait pu d'exploiter une mémoire partagée contenant la valeur de la marque. Dans ce cas MANGLE_1 ne modifierait pas le flux de données et se contenterait simplement de mettre à jour la valeur de la marque dans la mémoire partagée.

5.5.4.2 Détection d'anomalie

Si on applique cette modélisation par flux avec exploitation des CPN, l'analyse des conflits peut être réalisée. En effet en s'intéressant aux transformations successives du flux de données opérées par le passage dans les GAM successifs : MANGLE_1, ROUTER, MANGLE_2 et FILTER on peut déterminer si le flux va ou non passer d'un GAM à l'autre et donc conduire à une validation ou pas d'un enchainement de transformations.

On simule la transmission d'un trafic SMTP entrant dans R_1, comme un flux destiné à l'adresse IP destination « 192.168.2.2 » avec un numéro de port « 25 ». Le passage par la succession de GAM (MANGLE_1 → ROUTER → MANGLE_2), donne à l'attribut « *fwmark* » inséré par le pseudo protocole « mangle-mark » la valeur 2 lorsqu'il est transmis en fin de traitement à FILTER :

a) Dans la chaine PREROUTING, MANGLE_1 marque le bit b_0 à « 1 » et tous les autres bits à zéro,
b) Dans la chaine FORWARD, MANGLE_2 marque le bit b_0 à « 0 » (puisque b_0 = 1 et que $b_0 \leftarrow b_0$ XOR 1),
c) Par ailleurs MANGLE_2 marque le bit b_1 à « 1 » (trafic destiné à 192.168.2.2) et donc *fwmark = 0x0002*.

En conséquence, comme le flux possède une marque égale à « 2 » le paquet reçu va passer sur le réseau 2 (Figure 53).

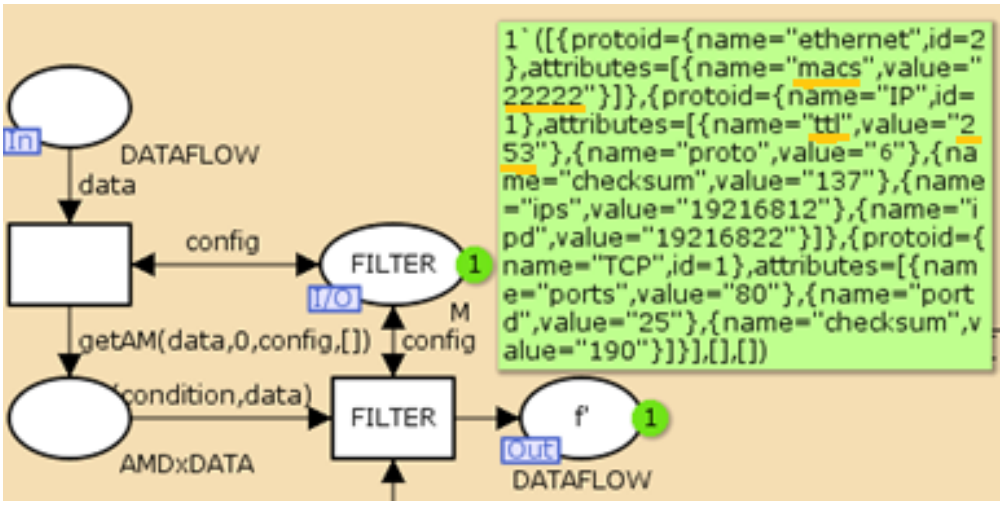

Figure 53. Flux de données acceptés par GAM Filter

Lors de l'injection d'un flux de données, d'adresse IP destination « 192.168.2.4 » et de port « 25 », le passage par la succession de GAM (MANGLE_1 → ROUTER → MANGLE_2) donnera à « *fwmark* » inséré par le pseudo protocole « mangle-mark », la valeur « 3 » avant d'être transmis à FILTER :

a) Dans la chaine PREROUTING, MANGLE_1 ne marque pas le bit b_0 car il ne s'agit pas d'un trafic SMTP,

b) Dans la chaine FORWARD, MANGLE_2 marque le bit b_0 à 1 car le numéro de port est inférieur à 1024

c) MANGLE_2 marque le bit b_1 à 1 car le système visé fait partie du réseau 2 (192.168.2.0) et donc *fwmark = 0x0003*

En conséquence, comme le flux possède une marque égale à « 3 » le paquet est rejeté par le module FILTER (Figure 54).

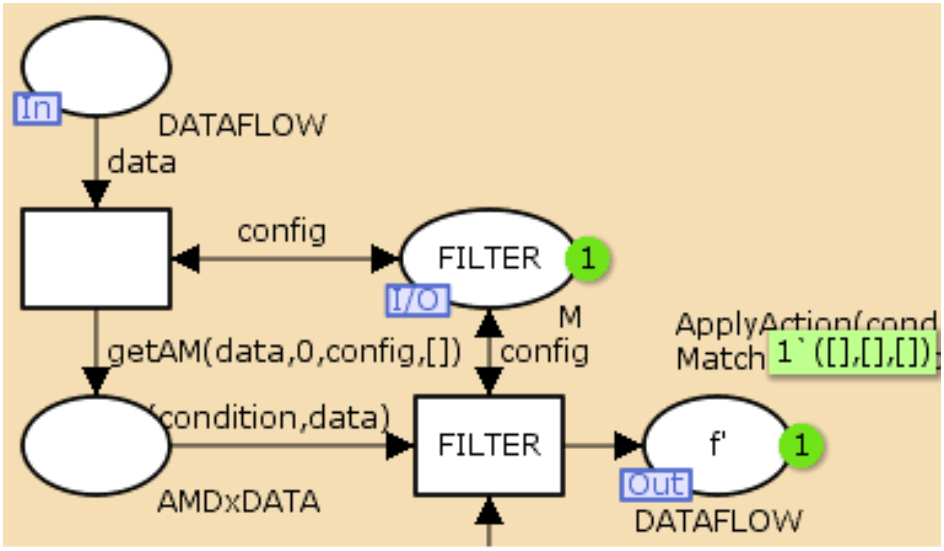

Figure 54. Flux de données supprimés par GAM Filter

L'administrateur découvre ainsi qu'un paquet dirigé vers l'adresse « 192.168.2.2 » avec le port de destination égal à « 25 » est transmis au lieu d'être rejetée alors qu'un paquet à l'adresse « 192.168.3.2 » avec le port de destination égale à « 25 » est rejeté au lieu d'être routé vers R_2.

5.6 Conclusion

Nous avons donné dans ce chapitre réservé aux expérimentations quatre scenarios avec GAM, qui nous permettent de porter du crédit à l'exploitation d'une modélisation reposant sur l'exploitation de flux pour valider ou invalider des configurations. Dans cette section les exemples ont porté sur des cas particuliers connus par les administrateurs (IPsec/NAPT) qui sont par essence représentatifs de situations où de l'existence de flux transmis d'un système à un autre. Dans les exemples présentés ici, les opérations de simulation réalisées nous ont conduit à démonter les possibilités de détection de blocage. En particulier les informations associées à la représentation d'un flux de données et à la constitution des ensembles d'authentification et de chiffrement « AUTHN et CONF » devraient permettre d'automatiser la détection de conflits lors de la composition d'opération de transformations [El Khoury et al. 2012 b].

Le premier scénario a présenté le conflit issu d'une composition de AH en mode tunnel et de NAPT, le deuxième démontre l'importance de l'ensemble AUTHN pour détecter l'incompatibilité de NAPT et de l'exploitation de AH en mode transport. Le troisième illustre la composition d'un chiffrement ESP et de NAPT illustre l'intérêt de l'existence des ensembles AUTHN et CONF.

Enfin le dernier exemple basé sur l'exploitation de Netfilter/iptables nous a permis de démonter qu'une information telle qu'une « marque iptables » peut être également appréhendée et considérée comme un attribut constitutif d'un flux d'information [El Khoury et al. 2013].

En s'appuyant sur notre formalisme, nous pouvons construire des réseaux de Pétri colorés qui reflètent le fonctionnement des opérations de transformation en chaine. L'analyse du comportement de ces réseaux de Pétri ouvre des champs d'investigation pour porter un regard critique sur l'impact de choix de configurations ou de reconfigurations. Ces résultats sont d'autant plus encourageants, car les conflits peuvent être détectés sans nécessiter de connaissance a priori ou d'expérience dans le domaine. Ceci nous pousse à croire que notre approche peut être utilisée pour détecter des conflits non connus impliquant de nouveaux mécanismes de sécurité pouvant agir à différents niveaux [El Khoury et al. 2013]. Ce travail nous permet de vérifier la capacité de notre modèle à exprimer et analyser les performances réelles et de discerner les inconsistances distribuées.

Chapitre 6. Conclusion générale et perspectives

6.1 Conclusion

La protection des données dans un environnement informatique est, depuis longtemps, reconnue/considérée comme un problème difficile. Ni le contrôle d'accès, ni le chiffrement, ne fournissent des solutions complètes pour assurer la confidentialité. Nous avons pu constater que le terme « politique de sécurité[8] » est primordial dans le contrôle de la sécurité d'un système informatique. Cette « politique » vise à garantir les services de sécurité (confidentialité, intégrité, disponibilité, traçabilité) qui doivent être implémentés et déployés selon des mécanismes existants (mécanisme de sécurité ou de contrôle d'accès). Notre travail de recherche a été motivé par le déploiement phénoménal aujourd'hui des processus dans le système d'information qui conduisent à une croissance continue des exigences en matière de sécurité.

De nombreux outils ont été proposés pour l'analyse et la détection des conflits liés au déploiement de la politique de sécurité. Bien que ces outils aient pu répondre à un ensemble d'exigences, ils demeurent efficaces uniquement dans un contexte spécifique dépendant des technologies développées et des exigences restrictives qui ont été émises lors de leur conception. En effet, beaucoup de modèles de gestion de l'information et d'analyse de la configuration de la sécurité des réseaux sont tout à fait adaptés pour un contrôle de bout en bout. Cependant, ils sont limités lorsqu'ils prennent en compte les équipements intermédiaires ou quand ils considèrent un grand nombre de mécanismes de sécurité, de multiples paramètres et une quantité innombrable de protocoles implémentés. Comment s'assurer d'une configuration homogène, correcte et non permissive ? Comment exprimer l'information de gestion tout en prenant en compte des contraintes comme l'hétérogénéité des solutions, les interdépendances entre les équipements, la pérennité du langage de modélisation face à l'évolution rapide des technologies ? Comment trouver le niveau d'abstraction adéquat pour, à la fois, être indépendant des mécanismes de sécurité et représenter fidèlement la réalité ?

Il est évident que ce type de problème est difficile à gérer [Laborde 2005] car il implique la prise en compte des dépendances entre des équipements et/ou des mécanismes complémentaires sur

[8] qui est l'expression des désirs en terme de sécurité d'un système donné.

des infrastructures distinctes ou/et virtuelles. Les solutions existantes sont souvent jugées insuffisantes pour :

- modéliser un flux d'une manière générique
- manipuler ce flux de données d'une manière unifiée
- modéliser un mécanisme avec sa configuration pouvant manipuler ces flux de données

La prise en compte de ces points devrait être formellement prouvée. Pour cela et avant de rechercher des méthodologies outillées de détection ou d'analyse, nous avons commencé par définir un modèle formel orienté flux de données indépendant de la technologie. Nous avons énoncé des concepts mathématiques clairs à implémenter dans un outil adapté. Nous avons alors pensé qu'il était possible d'atteindre une solution générique contrairement aux approches se focalisant sur la configuration d'un mécanisme où la plupart des travaux sont limités aux flux IP.

Le langage de spécification que nous avons adopté et qui a également été présenté dans [El-Khoury et al. 2011 a et b], a été le point de départ vers un modèle formel orienté flux de données pour l'analyse des politiques de sécurité réseau. Dans cette approche, un flux de données est représenté comme étant une suite d'éléments. Cette modélisation représente fidèlement un flux physique de données qui est une suite d'octets regroupés selon les spécifications des protocoles réseau. Cette représentation formelle est basée sur une abstraction d'un flux de données comprenant des blocs physiques, ainsi que la relation entre chaque bloc et le protocole sous-jacent.

D'autre part, les mécanismes de sécurité sont représentés sous forme de fonctions de transformation manipulant ces flux. La spécification de la configuration des mécanismes de sécurité est définie sous forme de boites noires réutilisables s'adaptant à chaque mécanisme en changeant leurs entrées et leurs sorties. Ce travail a été présenté dans [El-Khoury et al. 2012 a].

Pour valider notre travail, nous avons simulé notre langage de modéliastion en utilisant le logiciel « CPN Tools » [CPN Tools]. Le résultat de cette spécification est l'outil GAM-CPN (Generic Attribute-based Mechanism) qui est un modèle générique basé sur les attributs des flux de données pour représenter un mécanisme de sécurité quelconque. L'architecture de GAM-CPN est flexible et modulaire, elle simplifie la création d'un réseau formé de plusieurs mécanismes hétérogène de sécurité. Elle permet également l'observation et l'analyse de l'impact des fonctions de transformation (qui représentent les différents mécanismes de sécurité) sur les attributs du flux de données. Ce travail a été déjà publié dans [El-Khoury et al. 2012 b] pour démontrer la généralité et la simplicité de l'utilisation du modèle.

Le modèle GAM-CPN a été vérifié en simulant des cas réels, pour cela nous avons illustré quatre scénarii basés sur des technologies connues des administrateurs (par exemple, IPsec, NAPT et FW-iptables). Dans les trois premiers scenarii nous avons utilisé les informations associées à la représentation d'un flux de données et à la constitution des ensembles d'authentification et de

configuration pour pouvoir automatiser la détection de conflits lors de la composition d'opération de transformations [El-Khoury et al. 2012 a]. Le quatrième scénario relatif à la mise en œuvre de fonctionnalités « iptables » était également proposé afin de montrer la possibilité de détecter des anomalies qui peuvent naître de l'utilisation conjointe de mécanismes propres à la technologie iptables. En outre, nous avons démontré que des technologies plus complexes et alliant un degré d'abstraction plus fin peuvent être présentées par GAM-CPN. Ce travail a aussi été publié dans [El-Khoury et al. 2013].

6.2 Perspectives

Des contraintes appliquées aux flux de données et aux mécanismes de sécurité permettent de mettre en évidence les conflits grâce à notre démarche de modélisation qui serait aboutie si elle était outillée pour analyser les conflits **automatiquement**.

En effet, l'analyse fine des problèmes d'inconsistance vise à découvrir leur impact sur de nouveaux déploiements. Nous avons ainsi convergé vers un outil facilitant le travail de l'exploitant (administrateur) et du concepteur (ingénieur) tout en permettant à une tierce personne de prendre la bonne décision, de connaître à l'avance la faisabilité d'une configuration spécifique et de tester les équipements aux extrémités communicantes. Ceci donnera à notre analyse une vision de la corrélation entre les dispositifs, ainsi que l'effet produit par la sécurité globale et locale de chacun d'eux.

Les résultats sont encourageants et sont suffisamment généraux parce que la détection de conflits n'a pas nécessité de connaissances ou une expertise a priori. Cela nous fait croire que notre approche peut être utilisée pour détecter des conflits inconnus impliquant de nouveaux mécanismes de sécurité pouvant agir à différents niveaux. Et cela la rend une étape avancée dans le domaine d'évaluation de la configuration.

En résumé, il est important de définir une spécification générique pour configurer et représenter une technologie qui peut comporter un ou plusieurs mécanismes et peut jouer un rôle différent selon le niveau de protocole où elle est mise en œuvre.

Enfin, notre travail dans cette thèse s'est concentré sur l'analyse par simulation des conflits de sécurité. Il nous manque un outil qui permet d'analyser ces conflits automatiquement et toujours de manière exhaustive. Une piste de nos travaux futurs consiste à automatiser l'analyse en permettant aux intéressés de définir des propriétés en logique temporelle par exemple et de les contrôler automatiquement. L'objectif est de fournir un outil pouvant suivre un flux de données quelconque avec certaines caractéristiques à tout moment.

Acronymes

ACL : access control list

AD : Authentication Data

ADSL : Asymmetric Digital Subscriber Line

AEF : Active End-Flow

AES-CBC : Advanced Encryption Standard-cipher block chaining

AH : Authentication Header

API : Application Programming Interface

ARP : Address Resolution Protocol

AUTHN : ensemble d'authentification

BDD : diagrammes de décision binaires

CA : certification authority

CC : Critères Communs

CEI : Commission électrotechnique internationale

CONF : l'ensemble de confidentialité

CPN : Coloured Petri Nets

CRA : Conflict Resolution Algorithm

CTL : Computation tree logic

DAC : Discretionary Access Control

DAP : Directory Access Protocol

DES (3DES-CBC) : Data Encryption Standard (Triple Data Encryption Standard-cipher block chaining)

DMTF : Distributed Management Task Force

DMZ : Demilitarized Zone

DNS : Domain Name System

DOD : Department Of Defense

DSCP : Differentiated services code point

ECN: Electronic Communication Network

EF : End Flow (Terminaisons de flux)

ESP : Encapsulating Security Payload

FDD : firewall decision diagram

FIREMAN: FIREwall Modeling and ANalysis

FTP : File Transfer Protocol

GAM : Generic attribute-based mechanism model

GRE : Generic Routing Encapsulation

gw : gateway

HMAC : Hash Message Authentication Code

HTTP : HyperText Transfer Protocol

IAM : Identity Access Management

ICP : Infrastructures à Clés Publiques

IDS : Intrusion Detection System

IEEE : Institute of Electrical and Electronics Engineers

IETF : Internet Engineering Task Force

IKE : Internet Key Exchange

IMAP : Internet Message Access Protocol

IP : Internet Protocol

IPX : Internetwork Packet Exchange

IPv4 : Internet Protocol version 4

IPv6 : Internet Protocol version 6

IPsec : Internet Protocol Security

ISAKMP : Internet Security Association and Key Management Protocol

ISO : International Standard Organisation

ISMS : Information Security Management System

ITIL : IT Infrastructure Library

ITSEC : Information Technology Security Evaluation Criteria

ITU : International Telecommunication Union

L2F : Layer Two Forwarding

L2TP : Layer Two Tunneling Protocol

M : Mécanisme

MAC : Mandatory Access Control

MD5 : Message Digest 5

MIRAGE : MIsconfiguRAtion manaGEr

ML : metalanguage

MPLS : MultiProtocol Label Switching

NAPT : Network Address and Port Translation

NAT : Network address translation

NAT-T : Network address translation traversal

NIDS : Network Intrusion Detection System

NIST : National Institute of Standards and Technology

OSI : Open Systems Interconnection

OrBAC : organization-based access control

PCI : Protocol Control Information

PDP : Policy Decision Point

PDU : Protocol Data Unit

PEF : Passive End-Flow

PEP : Policy Enforcement Point

PKI : Public Key Infrastructure

PPP : Point-to-Point Protocol

PPPoA : point-to-point protocol over ATM

PPPoE : Point-to-point protocol over Ethernet

PPTP : Point-to-Point Tunneling Protocol

RBAC : Role Based Access Control

RPC : Remote Procedure Call

SDU : Service Data Unit

SHA-1 : Secure Hash Algorithm-1

SMSI : systèmes de management de la sécurité de l'information

SMTP : Simple Mail Transfer Protocol

SNAT : Source Network Address Translation

SOAP : Simple Object Access Protocol

SSH : Secure Shell

SSL : Secure Socket Layer

TCP : Transport Control Protocol

TCSEC : Trusted Computer Security Evaluation Criteria

TI : Technologies de l'Information

TLS : Transport Layer Security

TOE : Target Of Evaluation

TOS : Terms of service

TTL : Time to live

UDP : User Datagram Protocol

VLAN : Virtual Local Area Network

VPN : Virtual Private Network

VXLAN : Virtual eXtensible Local Area Networks

WEB : Wired Equivalent Privacy

XML : Extensible Markup Language

Références

[Abou El Kalam et al. 2003] Abou El Kalam A., El Baida R., Balbiani P., Benferhat S., Cuppens F., Deswarte Y., Miège A., Saurel C., Trouessin G., "Organization Based Access Control", IEEE 4th International Workshop on Policies for Distributed Systems and Networks (Policy 2003), 2003.

[Alfaro et al. 2008] Alfaro J., Cuppens N., Cuppens F., "Complete analysis of configuration rules to guarantee reliable network security policies", In Int. J. Inf. Secur. 7:103–122, 2008.

[Alfaro et al. 2010] Alfaro J., Cuppens N., Cuppens F. et S. Preda, "Mirage: a management tool for the analysis and deployment of network security policies," in Proc. of the 5th Int. Workshop on data privacy management, 2010, pp. 203–215.

[Al-Shaer et al. 2003] Al-Shaer E. et Hamed H. "Firewall policy advisor for anomaly discovery and rule editing". In Integrated Network Management, pages 17–30, 2003.

[Al-Shaer et al. 2004] Al-Shaer E. et Hamed H., "Discovery of Policy Anomalies in Distributed Firewalls", IEEE INFOCOM'04, 2004.

[Al-Shaer et al. 2005] Al-Shaer E. et Hamed H., " conflicts classification and analysis of distributed firewall policies". pages 2069,2085. IEEE J.Select Areas Common. Magazine, 2005.

[Al-Shaer et al. 2006] Al-Shaer E. et Hamed H., "Taxonomy of conflicts in network security policies". pages134–141. IEEE Common. Magazine, 2006.

[Al-Shaer et al. 2009] Al-Shaer E., Marrero W., El-Ataw, A., ElBadawi K., "Network configuration in a box: towards end-to-end verification of network reachability and security", Network Protocols, 2009. ICNP 2009. 17th IEEE International Conference on , vol., no., pp.123-132, 13-16 Oct. 2009.

[Al-Shaer et al. 2010] Al-Shaer E., AL-Haj S., "FlowChecker: Configuration Analysis and Verification of Federated OpenFlow Infrastructures", SafeConfig' 10, October 4, 2010, Chicago, Illinois, USA. Copyright 2010 ACM 978-1-4503-0093-3/10/10

[Andys] URL: http:www.andys.org.uk/bits/2010/01/27/iptables-fun-with-mark

[Abrial 1996] Abrial, J.R., "The B-Book — Assigning Programs to Meanings", Cambridge University Press, Cambridge (1996).

[Avizienis et al. 2004] Avizienis A., Laprie J., Randell B. et Landwehr C., "Basic Concepts and Taxonomy of Dependable and Secure Computing", IEEE Transactions on Dependable and Secure Computing, vol. 1, pp. 11-33, 2004.

[Atelier B] URL: http://www.atelierb.eu/html/produit_en.html/.

[Bartal et al. 1999] Bartal Y., Mayer A., Nissim K., Wool A., "FIRMATO : A Novel Firewall Management Toolkit", in 20th IEEE Symposium on Security and Privacy, pages 17–31, Oakland, California, May 1999.

[Basile et Lioy 2004] Basile C. et Lioy A., "Towards an algebraic approach to solve policy conflicts", in Workshop on Logical Found. of an Adaptive Security Infrastructure (WOLFASI), July 2004.

[Basile et al., 2012] Basile C, Cappadonia A, Lioy A., "Network-Level Access Control Policy Analysis and Transformation", IEEE/ACM Trans. Netw. 20(4): 985-998 (2012)

[Benaissa et al. 2006] Benaissa, N., Cansell, D., M´ery D., "Integration of Security Policy into System Modeling", LNCS, vol. 4355, pp. 232–247. Springer, Heidelberg (2006)

[Bishop, 2002] Bishop, M., “Computer Security - Art and Science”. Addison-Wesley Professional (December 12, 2002). ISBN-10: 0201440997.

[Buttyan et al. 2009] Buttyán L., Pék G. et Thong T. , “Consistency verification of stateful firewalls is not harder than the stateless case”, Infocommunications Journal, LXIV(2009/2-3), 2-3 2009.

[B2RODIN] URL: http://www.methode-b.com/en/tools/rodin/b2rodin/

[Castagnetto et al. 1999] Castagnetto J. et al., « Professional PHP Programming.Wrox Press Inc », ISBN 1-86100-296-3, 1999

[CC 1999a] Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, 60 p., ISO/IEC 15408-1 (1999).

[CC 1999b] Common Criteria for Information Technology Security Evaluation, Part 4: Predifined Protection Profiles, 166 pp., ISO/IEC 15408-1 (1999).

[Cheaito 2012] Cheaito M., « Un cadre de spécification et de déploiement de politiques d’autorisation » Thèse soutenu en 2012.

[Chinaei et al. 2007] Chinaei A., Chinaei H., Tompa F., “A Unified Conflict Resolution Algorithm”, W. Jonker and M. Petković (Eds.): SDM 2007, LNCS 4721, pp. 1–17, 2007. © Springer-Verlag Berlin Heidelberg 2007

[CPN Tools] URL : http://cpntools.org/

[Cuppens et al. 2005a] Cuppens F., Cuppens N., Alfaro J., “Detection and removal of firewall misconfiguration”, 2005 IASTED International Conference on Communication, Network and Information Security (CNIS 2005). Phoenix, AZ, USA, Novembre, 2005.

[Cuppens et al. 2005b] Cuppens F., Cuppens N., Alfaro J., “Misconfiguration Management of Network Security Components”, Proceedings of the 7th International Symposium on System and Information Security, Sao Paulo, 2005

[Cuppens et al. 2012] Cuppens F., Cuppens N., Alfaro J., Moataz T., Rimasson X., “Handling Stateful Firewall Anomalies”, 27th IFIP TC 11 Information Security and Privacy Conference, SEC 2012, Heraklion, Crete, Greece, June 4-6, 2012. pp 174-186

[El-Khoury et al. 2011a] El Khoury H., Laborde R., Barrère F., Benzekri A., “Towards a Formal Data Flow Oriented Model for Network Security Policies Analysis”, in SAR-SSI, p. 1-7, 2011.

[El-Khoury et al. 2011b] El Khoury H., Laborde R., Barrère F., Benzekri A., Chamoun M., “A Generic Data Flow Security Model” (poster). Symposium on Configuration Analytics and Automation (SafeConfig 2011), Arlington, VA, USA, 31/10/2011-01/11/2011, IEEE, p. 1-2, 2011.

[El-Khoury et al. 2012a] El Khoury H., Laborde R., Barrère F., Benzekri A., Chamoun M., “A Formal Data Flow-Oriented Model For Distributed Network Security Conflicts Detection”. International Conference of Networking and Services (ICNS 2012), St. Maarten, The Netherlands Antilles, 25/03/2012-30/03/2012, Xpert Publishing Service (XPS), p. 20-27, 2012.

[El-Khoury et al. 2012b] El Khoury H., Laborde R., Barrère F., Benzekri A., Chamoun M., “A Generic Attribute-Based Model for Network Security Mechanisms Representation and Configuration”, in International Conference on Frontier of Computer Science and Technology (FCST 2012), Suzhou, China, 21/11/2012-23/11/2012, Soochow University, novembre 2012.

[El-Khoury et al. 2013] El Khoury H., Laborde R., Barrère F., Benzekri A., Chamoun M., “A Specification Method for Analyzing Fine Grained Network Security Mechanism Configurations”(short paper). Symposium on Configuration Analytics and Automation (SafeConfig 2013), Washington, D.C., USA, 16/10/2013, (Eds.), IEEE, p. 483-487, 2013.

[Ferraiolo et al. 2001] Ferraiolo, David F., Sandhu R., Gavrila S., Kuhn D. R. et Chandramouli R., “Proposed NIST standard for role-based access control”, ACM Transactions on Information and System Security (TISSEC) 4, no. 3 (2001): 224-274.

[Fisch et White 1999] Fisch E., White G., “Secure Computers and Networks: Analysis, Design, and Implementation”, ISBN-10: 0849399939 and ISBN-13: 978-0849318689, December 28, 1999.

[frozentux] URL: https://www.frozentux.net/iptables-tutorial/iptables-tutorial.html#IPFILTERING

[Fu et al. 2001] Fu Z., Wu F., Huang H., Loh K., Gong F., Baldine I., Xu C., "IPSec/VPN Security Policy: Correctness, Conflict Detection and Resolution", Proceedings of IEEE POLICY, 2001.

[Gouda et al. 2005] Gouda M. et Liu A., "A Model of Stateful Firewalls and its Properties", Proceedings of the 2005 International Conference on Dependable Systems and Networks (DSN'05) 0-7695-2282-3/05

[Gouda et al. 2007] Gouda M. et Liu A., "Structured firewall design", The International Journal of Computer and Telecommunications Networking, Volume 51 Issue 4, March, 2007 Pages 1106-1120.

[Guttman et Herzog 2003] Guttman J., Herzog A., "Rigorous automated network security management", Technical report, MITRE Corp., Aug. 2003. Preliminary version appeared in Proc. VERIFY 2002.

[Guttman et Herzog 2005] Guttman J., Herzog A., "Rigorous automated network security management", International Journal of Information Security, Issue 3, Volume 4, 2005.

[Harper 2011] Harper R., « Programming in Standard ML », Carnegie Mellon University. Springer Semester, 2011.

[ISO 15408] Common Criteria for Information Technology Security Evaluation, norme ISO/CEI 15408, version 2.1, août 1999.

[ISO 7498-1] International Standard Organisation, « Information technology-Open Systems Interconnection-Basic Reference Model: The Basic Model », ISO 7498-1, 1994.

[ISO 7498-2] International Standard Organisation, « Information Processing Systems, Open Systems Interconnection Reference Model - Security Architecture », ISO 7498-2, Juillet 1988.

[ISO 7498-4] ISO, "Management Framework", ISO 7498-4, 1989.

[ISO/IEC 27000] URL : https://www.iso.org/obp/ui/#iso:std:iso-iec:27000:ed-3:v1:en

[ITSEC 1991] ITSEC , Information Technology Security Evaluation Criteria, v 1.2, 136 pp., ISBN 92-826-3005-6, Office des publications officielles des Communautés Européennes, Luxembourg, 1991.

[Jensen et Kristensen 2009] Jensen K., Kristensen L. "Coloured Petri Nets: Modelling and Validation of Concurrent Systems", ISBN 978-3-642-00283-0 and e-ISBN 978-3-642-00284-7 Springer-Verlag Berlin Heidelberg 2009

[Kamel] Kamel M., «Patrons organisationnels et techniques pour la sécurisation des Organisations Virtuelles » Thèse soutenu en 2008.

[Laborde et al. 2004a] Laborde R., Nasser B., Grasset F., Barrère F., Benzekri A., "Une nouvelle technique pour l'évaluation formelle de mécanismes de sécurité réseaux", 3ème Conférence sur la Sécurité et Architectures Réseaux (SAR), 2004.

[Laborde et al. 2004b] Laborde R., Nasser B., Grasset F., Barrère F., Benzekri A., "A formal approach for the evaluation of network security mechanisms based on RBAC policies. " Electronic Notes in Theoretical Computer Science, Vol. 121, Elsevier, proceedings of WISP'04, 2004.

[Laborde et al. 2004c] Laborde R., Nasser B., Grasset F., Barrère F., Benzekri A., "A formal tool for user based network security policy specification ", International Workshop Security Analysis of Systems: Formalisms and Tools, 2004.

[Laborde et al. 2004d] Laborde R., Nasser B., Grasset F., Barrère F., Benzekri A., "Network Security Management: A Formal Evaluation Tool based on RBAC Policies", IFIP NetCon, Springer ISBN 0-387-23197-8, 2004.

[Laborde 2005] Laborde R., « Un Cadre formel pour le raffinement de l'information de gestion de sécurité réseau : Expression et Analyse » Thèse soutenu en 2005.

[Laborde et al. 2007] R. Laborde, M. Kamel, F. Barrére, and A. Benzekri, "Implementation of a Formal Security Policy Refinement Process in WBEM Architecture", Journal of Network and Systems Management, 15(2), 2007.

[Lastera 2014] Lastera M., « Architecture securisée pour les systémes d'information des avions du futur ». Thèse soutenu en 2014.

[Laprie et al. 1995] Laprie J.C., Arlat J., Blanquart J.P., Costes A., Crouzet Y., Deswarte Y., Fabre J.C., Guillermain H., Kaâniche M., Kanoun K., Mazet C., Powell D., Rabéjac C. et Thévenod P., « Guide de la sûreté de fonctionnement », 324 pp., Editions Cépaduès, Toulouse 1995.

[Lee et al. 2002] Lee T.K., Yusuf S., Luk W., Sloman M., Lupu E., Dulay N., "Development Framework for Firewall Processors", Field-Programmable Technology, 2002. (FPT). Proceedings. 2002 IEEE International Conference.

[Microsoft-1] URL : http://msdn.microsoft.com/en-us/library/dd340931.aspx

[Microsoft-2] URL : http://msdn.microsoft.com/en-us/library/hh780717.aspx

[Microfocus] URL :http://documentation.microfocus.com/help/

[Msexchange] URL :http://www.msexchange.org/articles-tutorials/exchange-server-2003/migration-deployment

[Netfilter] URL: http://ipset.netfilter.org/iptables-extensions.man.html

[NIST 1995] Guttman B. et Roback E., "An Introduction to Computer Security: The NIST Handbook" U.S. DEPARTMENT OF COMMERCE, Technology Administration National Institute of Standards and Technology, 1995.

[Preda et al. 2010] Preda S., Cuppens N., Cuppens F., Alfaro J. et Toutain L., "Model-Driven Security Policy Deployment: Property Oriented Approach". ESSoS 2010, LNCS 5965, pp. 123–139, 2010.

[RFC 790] Postel J., « Assigned Numbers », IETF RFC 790, 1981

[RFC 1155] Rose M., McCloghrie K., "Structure and Identification of Management Information for TCP/IP-based Internets", IETF RFC 1155, 1990.

[RFC 1851] Karn P., Metzger P., Simpson W., "The ESP Triple DES Transform", IETF RFC 1851, 1995.

[RFC 1918] Rekhter Y., Moskowitz B., Karrenberg D., J. de Groot G., Lear E., "Address Allocation for Private Internets", IETF RFC 1918, 1996.

[RFC 2104] Krawczyk H., Bellare M., Canetti R., "HMAC", IETF RFC 2104, 1997.

[RFC 2251] Wahl M., Howes T., Kille S., "Lightweight Directory Access Protocol (v3)", IETF RFC 2251, 1997.

[RFC 2401] Kent S., Atkinson R., "Security Architecture for the Internet Protocol", IETF RFC 2401, 1998.

[RFC 2402] Kent S., Atkinson R., "IP Authentication Header (AH)", IETF RFC 2402, 1998.

[RFC 2403] Madson C., Glenn R., "The Use of HMAC-MD5-96 within ESP and AH", IETF RFC 2403, 1998.

[RFC 2404] Madson C., Glenn R., "The Use of HMAC-SHA-1-96 within ESP and AH", IETF RFC 2404, 1998.

[RFC 2406] Kent S., Atkinson R., "IP Encapsulating Security Payload (ESP)", IETF RFC 2406, 1998.

[RFC 2451] Pereira R., Adams R., "The ESP CBC-Mode Cipher Algorithms", IETF RFC 2406, 1998

[RFC 2784] Farinacci D., Hanks S., Meyer D., Traina P. "Generic Routing Encapsulation (GRE)", IETF RFC 2784, 2000

[RFC 2516] Mamakos L., Lidl K., Evarts J., Carrel D., Simone D., Wheeler R., "A Method for Transmitting PPP Over Ethernet (PPPoE)", IETF RFC 2516, 1999

[RFC 2578] McCloghrie K., Perkins D., Schoenwaelder J., "Structure of Management Information Version 2 (SMIv2)", IETF RFC 2578, 1999.

[RFC 2637] Hamzeh K., Pall G., Verthein W., Taarud J., Little W., Zorn G.,"Point-to-Point Tunneling Protocol (PPTP)", IETF RFC 2637, 1999.

[RFC 2661] Townsley W., Valencia A., Pall G., Zorn G., Palter B.," Layer Two Tunneling Protocol (L2TP)", IETF RFC 2661, 1999.

[RFC 2784] Boyle J., Cohen R., Durham D., Herzog S., Rajan, R. and A. Sastry, "The COPS(Common Open Policy Service) Protocol", IETF RFC 2784, 2000.

[RFC 2753] Yavatkar R., Pendarakis D., Guerin R., "A Framework for Policy-based Admission Control", IETF RFC 2753, 2000.

[RFC 2828] Shirey R., "Internet Security Glossary", IETF RFC 2828, 2000.

[RFC 2989] Aboda B., Calhoun P., Glass S., Hiller T., McCann P., Shiino H., Walsh P., Zorn G, Dommety G., …, "Criteria for Evaluating AAA Protocols for Network Access", IETF RFC 2989, 2000.

[RFC 3022] Srisuresh P., Egevang K., “Traditional IP Network Address Translator (Traditional NAT)”, IETF RFC 3022, 2001.

[RFC 3084] Chan K., Seligson J., Durham D., Gai S., McCloghrie K., Herzog S., Reichmeyer F., Yavatkar R., Smith A., “COPS Usage for Policy Provisioning (COPS-PR)”, IETF RFC 3084, 2001.

[RFC 3159] McCloghrie K., Fine M., Seligson J., Chan K., Hahn S., Sahita R., Smith A., Reichmeyer F., “Structure of Policy Provisioning Information (SPPI)”, IETF RFC 3159, 2001.

[RFC 3216] Elliott C., Harrington D., Jason J., Shoenwaelder J., Strauss F., Weiss W., “SMIng Objectives”, IETF RFC 3216, 2001.

[RFC 3394] Schaad J., Housley R., “Advanced Encryption Standard (AES) Key Wrap Algorithm”, IETF RFC 3394, 2002.

[RFC 3460] Moore B., “Policy Core Information Model (PCIM) extensions”, IETF RFC 3460, 2003.

[RFC 3602] Frankel S., Glenn R., Kelly S., “The AES-CBC Cipher Algorithm and Its Use with IPsec”, IETF RFC 3602, 2003.

[RFC 3715] Aboba B., Dixon W., “IPsec-Network Address Translation (NAT) Compatibility Requirements”, IETF RFC 3715, 2004.

[RFC 4253] Ylonen T., Lonvick C., “The Secure Shell (SSH) Transport Layer Protocol”, IETF RFC 4253, 2006.

[RFC 4301] Kent S., Seo K., “Security Architecture for the Internet Protocol”, IETF RFC 4301, 2005.

[RFC 4302] Kent S., “IP Authentication Header (AH)”, IETF RFC 4302, 2005.

[RFC 4303] Kent S., “IP Encapsulating Security Payload (ESP)”, IETF RFC 4303, 2005.

[RFC 4304] Kent S., “(ESN) Addendum to IPsec (DOI) for (ISAKMP)”, IETF RFC 4304, 2005.

[RFC 4305] Eastlake 3rd D., “Cryptographic Algorithm Implementation Requirements for (ESP) and (AH)”, IETF RFC 4305, 2005.

[RFC 4306] Kaufman C., “Internet Key Exchange (IKEv2) Protocol”, IETF RFC 4306, 2005.

[RFC 4307] Schiller J., “Cryptographic Algorithms for Use in the (IKEv2)”, IETF RFC 4307, 2005.

[RFC 4308] Hoffman P., “Cryptographic Suites for IPsec”, IETF RFC 4308, 2005.

[RFC 4309] Housley R., “Using (AES) CCM Mode with IPsec (ESP)”, IETF RFC 4309, 2005.

[RFC 4949] Altman J., “Telnet Encryption: CAST-128 64 bit Output Feedback ”, IETF RFC 4949, 2000.

[RFC 5246] Dierks T., Rescorla E., “The Transport Layer Security (TLS) Protocol”, IETF RFC 5246, 2008.

[RFC 6101] Freier A., Karlton P., Kocher P., “The Secure Sockets Layer (SSL) Protocol”, IETF RFC 6101, 2011.

[Samarati et al. 2001] Samarati P., De Capitani di Vimercati S., “Access Control: Policies, Models and Mechanisms”, Foundations of Security Analysis and Design, LNCS 2171, Springer-Verlag. 2001.

[Sandhu et al., 1996] Sandhu R., Coyne E. J., Feinstein H. L. et Youman, C. E, “Role-based access control models”. In IEEE Computer, volume 29(2), pages 38–47

[Sandhu et Samarati 1994] Sandhu R. et Samarati P., « Access control: principles and practice ». Dans IEEE Communications Magazine, 1994.

[Scottlowe] URL: http://blog.scottlowe.org/2013/05/29/a-quick-introduction-to-linux-policy-routing/

[SecurityInfo] URL : http://www.securiteinfo.com/conseils/nat.shtml

[TCSEC 1985] Departement of Defense, “Trusted Computer System Evaluation Criteria”, DoD 5200.28-STD, Décembre 1985.

[Uribe et Cheung 2007] Uribe T. E., Cheung S., “Automatic analysis of firewall and network intrusion detection system configurations”, Journal of Computer Security, 15(2007) 691-715, IOS Press, 2007.

[Web Services 2008] Michael P. P., “Web Services: Principles & Technology”, 1st Edition, © Pearson Education Limited 2008

[William Stalling 2011] Stalling W., “Network security essentials: applications and standards” - fourth edition, Pearson Education, 2011.

[wireshark] URL : http://www.wireshark.org

[xmlenc] URL : http://www.w3.org/TR/xmlenc-core/

[X.500] ITU-T recommendations, “Information technology - Open Systems Interconnection - The Directory: Overview of concepts, models and services”, X.500, 2012. Disponible à : https://www.itu.int/rec/T-REC-X.500-201210-I/en

[X.800] ITU-T recommendation, “Security architecture for Open Systems Interconnection for CCITT applications”, X.800, 1991. Disponible à : https://www.itu.int/rec/T-REC-X.800-199103-I/en

[X.805] ITU-T recommendation, “Security architecture for systems providing end-to-end communications”, X.805, 2003. Disponible à : https://www.itu.int/rec/T-REC-X.805-200310-I/en.

[Yuan et al. 2006] Yuan L., Mai J., Su Z., Chen H., Chuah C., Mohapatra P.,“FIREMAN: A toolkit for firewall modeling and analysis”. In IEEE Symposium on Security and Privacy, pp.199–213, 2006.