Tables des matières

Liste des figures

Liste des figures

Liste des tableaux

CHAPITRE I

Introduction générale

De nos jours, l'avènement de l'Internet des Objets (IoT) a changé le monde. L'IoT a permis l'émergence d'objets hétérogènes dans notre vie quotidienne. Ces objets ont désormais la capacité de se connecter à Internet et de communiquer sur le web. Des objets aussi communs qu'une ampoule, un compteur d'électricité, une pièce automobile, une machine à café ou un smartphone échangent des données sur Internet. Grâce à l'IoT, un objet intelligent est devenu un acteur de processus. Le nombre d'objets potentiels pouvant être connectés à l'IoT est de plusieurs centaines de milliards. Ces objets autonomes sont dotés de mémoires, d'un canal de communication, d'un processeur et de capteurs ou actionneurs conférant une forme d'intelligence. Les réseaux de capteurs sans fil (WSN) sont une technologie maîtresse qui a contribué au succès de l'IoT. Les WSNs visent à connecter le monde virtuel et le monde physique, où les objets peuvent communiquer de manière autonome entre eux et avec des systèmes intelligents. Pour garantir cette communication Internet, une adaptation des technologies et d'infrastructures IP aux contraintes des WSNs est strictement nécessaire. Le protocole Internet version 6 (IPv6) est considéré comme un candidat idéal pour l'IoT. Son espace d'adressage large convient au nombre croissant des dispositifs IoT. Le protocole IPv6 offre une interopérabilité, une évolutivité, une programmation facile et élimine le besoin des passerelles complexes. La standardisation d'IPv6 sur un réseau personnel à faible puissance (6LoWPAN) permet d'utiliser efficacement le protocole IPv6 dans les WSNs à ressources limitées. Ces réseaux sont appelés les réseaux 6LoWPANs. Ce sont des réseaux hybrides qui permettent l'interconnexion entre l'Internet et les appareils utilisant l'IPv6, ce qui forme l'IoT basé sur IP (IP-based IoT). Cette thèse se concentre sur l'IoT formé par l'interconnexion des réseaux WSN connectés à IP (WSNs IP- connectés) et Internet formant ainsi les réseaux 6LoWPANs.

Les nœuds utilisant 6LoWPAN communiquent directement avec des hôtes compatibles avec le protocole IPv6. Les hôtes IPv6 utilisent le protocole de sécurité Internet (IPSec) pour sécuriser leurs communications. Lorsque ces données circulent entre des hôtes IPv6 et des nœuds 6LoWPAN, il est souhaitable de tirer parti des avantages de sécurités offertes par IPsec. Le protocole IPsec est un protocole qui opère au niveau de la couche réseau du modèle OSI. Il permet de sécuriser les communications dans des réseaux non-

fiables tels que les réseaux locaux (LANs) et les réseaux étendus (WANs). Le protocole IPsec assure l'intégrité, l'authentification de la source et la confidentialité des données sans aucune modification sur les réseaux existants. Il fonctionne selon deux modes différents : le mode transport qui fournit une sécurité de bout en bout, et le mode tunnel utilisé dans les configurations entre passerelles ou bien entre hôtes et passerelles. IPsec fournit les services de sécurité à travers deux extensions d'en-tête, appelées : Authentication Header (AH) et Encapsulating Security Payload (ESP). Le type de traitement à appliquer sur un paquet IP est défini dans une politique de sécurité (SP). Une SP décide si un paquet doit être protégé par IPsec, passé sans protection ou bien écarté. Dans le cas de protection IPSec, le traitement nécessaire est définis dans une association de sécurité (SA). IPsec utilise un protocole d'échange de clé Internet (IKE) pour négocier les associations de sécurité d'une manière dynamique et sécurisée. Le protocole IKE est le cœur de l'architecture d'IPsec. Les exigences de sécurité fondamentales du protocole IPsec dépendent essentiellement d'IKE.

Dans cette thèse, nous visons à exploiter les avantages du protocole IPsec dans un réseau dynamique. Par conséquent, nous nous concentrons sur l'utilisation d'IPsec pour sécuriser les communications de bout en bout dans les réseaux 6LoWPANs. Nous travaillons sur un réseau 6LoWPAN comme étant un réseau personnel qui fait partie du domaine de l'IoT tel qu'un réseau domestique ou un smart home.

1.1 La problématique

6LoWPAN a essentiellement révolutionné le paysage de l'IoT en étendant l'utilisation de l'IPv6 aux objets intelligents et minuscules. L'IPv6 offre l'interconnexion de la plupart des objets physiques avec Internet. Cependant, cette ouverture sur le monde d'Internet, ouvre également la possibilité aux attaques et aux violations de sécurité. Le problème de sécurité des réseaux 6LoWPANs est un sujet controversé dans l'IoT. 6LoWPAN n'authentifie pas le nœud avant de rejoindre le réseau. Par conséquent, un attaquant peut facilement rejoindre le réseau et effectuer des attaques. D'où, la fourniture d'un mécanisme de confidentialité et de cryptage est primordiale pour atténuer les différentes attaques sur ces réseaux. Étant obligatoire dans IPv6, IPsec est approprié pour garantir la

sécurité de bout en bout dans les 6LoWPANs. Il nécessite relativement peu d'efforts pour activer IPsec sur les hôtes IPv6, car il est déjà implémenté au niveau de la couche réseau. IPsec, étant une solution standard pour Internet, assure la confidentialité, l'intégrité et l'authenticité. Ainsi, il couvre les exigences de sécurité de base de l'IoT. Cependant, IPSec est un protocole lourd et compliqué par rapport à la nature contraignante des dispositifs IoT. Pour cela, les travaux en cours recommandent l'utilisation d'une version compressée d'IPsec qui convient à ce type de réseaux. L'IPsec compressé assure toutes les capacités de sécurité du protocole tout en réduisant la taille des entêtes. L'utilisation d'IPsec compressé dans des environnements dynamiques nécessite une certaine flexibilité. La politique de sécurité est le premier élément dans l'architecture IPsec qui définit le traitement approprié sur les paquets IP échangés. Le processus de la gestion des politiques doit être assez dynamique est flexible pour convenir à la nature dynamique de l'IoT. De plus, cette flexibilité doit être aussi assurée au niveau de l'association de sécurité. Le protocole IKE assure ce processus d'une manière dynamique et sécurisée. Cependant, l'utilisation du protocole IKE associé à IPsec ne convient pas aux appareils avec contraintes. Outre ses nombreux avantages, IKEv2 souffre de certaines lacunes de sécurité, telles que sa vulnérabilité contre les attaques DoS et MITM. Par conséquent, un protocole d'authentification et d'échange de clé conçu particulièrement pour l'IoT est strictement nécessaire.

1.2 Objectifs du travail

L'objectif de notre thèse est d'exploiter le protocole IPsec dans un environnement dynamique distribué et hétérogène, tout en assurant un degré élevé de sécurité. Dans ce travail, nous visons à utiliser l'IPsec pour sécuriser les communications de bout en bout dans les réseaux 6LoWPANs. Nous concentrons notre travail sur deux éléments essentiels dans l'architecture de sécurité IPsec : la politique de sécurité et l'association de sécurité. En ce qui concerne la politique de sécurité, notre but est de fournir un mécanisme de gestion des politiques adapté aux environnements dynamiques. En général, la gestion des politiques est un problème critique. Ce problème s'aggrave encore plus lors de l'utilisation d'IPsec dans des environnements dynamiques. La nature de ces environnements et leurs contraintes présentes des défis majeurs à la gestion efficace des

politiques de sécurité. De ce fait, notre objectif est de fournir un moyen efficace de gestion de politique de sécurité tout en éliminant les conflits qui peuvent apparaître suite aux erreurs de configurations. Le deuxième élément sur lequel nous travaillons, est l'association de sécurité. Les associations de sécurité IPsec sont manipulées et échangées par le protocole IKE. Notre objectif est de concevoir un nouveau protocole d'authentification et d'échange de clé qui remplace l'IKE dans le protocole IPsec. Le nouveau protocole doit être mieux adapté aux dispositifs de l'IoT.

1.3 Contributions

Ce travail est composé de deux parties principales. La première concerne la gestion automatique et efficace des politiques de sécurité IPsec. Les contributions de cette partie sont résumées comme suit :

- Une classification des différents types de conflits qui peuvent exister dans une politique de sécurité IPsec.
- Une méthode de résolution des conflits qui consiste à donner le privilège aux règles ayant la priorité la plus élevée. La méthode proposée nommée, préséance pour la règle la plus prioritaire, en anglais Precedence for the Highest Priority Rule (PHPR). Cette méthode réduit le besoin d'intervention de l'administrateur pour résoudre les conflits [C1].
- Proposition d'un nouvel algorithme de génération automatique des politiques de sécurité IPsec sans conflits (Automatic Generation of Conflict-free IPsec Policy : AGCSP-IPsec) [J2]. L'algorithme proposé vise à éliminer les conflits d'une politique de sécurité automatiquement, tout en minimisant le temps de traitement. En outre, une implémentation de l'algorithme est faite pour démontrer, d'une part, son efficacité et, d'autre part, ses avantages par rapport à d'autres algorithmes similaires.

Dans la deuxième partie de notre travail, nous proposons un protocole léger d'authentification et d'échange de clés pour l'IoT (Tiny Authentication and Key Exchange protocole for IoT : TAKE-IoT) [J1]. Le protocole proposé vise à remplacer IKEv2. TAKE-IoT fonctionne avec IPsec pour assurer une gestion dynamique et

sécurisée des clés et d'association de sécurité entre les nœuds capteurs et les hôtes IPv6. En premier lieu, nous focalisons sur la complexité et les performances du protocole proposé afin de garantir le minimum cout de calcul et de consommation d'énergie. Pour cela, nous utilisons des techniques cryptographiques légères. Nous réduisons également le nombre de messages échangés ainsi que le nombre d'opérations effectuées lors de la génération des clés. En plus, nous concentrons nos efforts sur l'amélioration de sécurité de notre proposition. Notre protocole est plus sécurisé que le protocole IKEv2 conventionnel, il peut résister aux attaques les plus communes sur les réseaux WSNs, comme les attaques par déni de service, les attaques par rejeu et l'attaque homme au milieu.

1.4 Structure de la thèse

La suite de ce manuscrit est structurée comme suit :

Chapitre II : ce chapitre définit le paradigme de l'IoT basé sur IP et présente sa technologie sous-jacente, qui permet l'adaptation d'IPv6 dans les réseaux sans fil personnels à faible puissance (6LoWPAN). Il définit les réseaux 6LoWPANs, leur architecture, leur fonctionnement et leurs domaines d'application. Enfin, il met en évidence les différents objectifs et les défis de sécurité dans les réseaux 6LoWPAN, ainsi que les techniques proposées dans la littérature pour faire face à ces défis**.**

Chapitre III : ce chapitre présente le Protocole de Sécurité Internet IPsec. Premièrement, il présente les services de sécurité offerts par le protocole. Ensuite, il définit ses modes de fonctionnement. Il présente également son architecture de sécurité, en définissant les éléments essentiels qui fournissent les différents services de sécurité. Deuxièmement, ce chapitre introduit une étude des travaux proposés dans la littérature pour la vérification des politiques de sécurité. Enfin, il présente le protocole IKE utilisé par IPsec pour la gestion des clés et des associations de sécurité.

Chapitre VI : ce chapitre présente un état de l'art des solutions proposées dans la littérature pour l'authentification ainsi que la gestion des clés pour l'IoT. De plus, il présente les différents concepts de base de la cryptographie qui ont été étudiés dans le

cadre de cette thèse, y compris les clés cryptographiques et les algorithmes recommandés pour l'IoT.

Chapitre V : ce chapitre présente notre première contribution pour la gestion automatique des politiques de sécurité IPsec. Il présente notre algorithme de génération automatique des politiques de sécurité IPsec sans conflits (AGCSP-IPsec). De plus, il définit la méthode proposée pour détecter les conflits ainsi que les mécanismes utilisés pour les résoudre. Finalement, le chapitre présente une évaluation des performances de l'algorithme proposé et une comparaison avec les travaux existants.

Chapitre VI : ce chapitre introduit notre deuxième contribution concernant le protocole IPsec. Il présente notre protocole d'authentification et d'échange de clé pour l'IoT (TAKE-IoT). Il définit les objectifs de recherche que notre proposition doit accomplir. Finalement, le chapitre présente une analyse de sécurité du protocole proposé et une comparaison avec les mécanismes relatifs, existants dans la littérature.

CHAPITRE II

L'Internet des objets basé sur IP: fonctionnement et sécurité

L'objectif fondamental de ce travail est d'exploiter le protocole IPsec dans un environnement dynamique. L'environnement sur lequel nous avons travaillé est l'Internet des objets basé sur IP, en anglais : IP-based IoT. Dans ce chapitre, nous définissons le paradigme de l'IP-based IoT. Nous introduirons également sa technologie sous-jacente : 6LoWPAN. Cette dernière permet l'adaptation du protocole IPv6 dans les réseaux personnels sans fil à faible puissance (IPv6 for Low Power Personal Area Network : 6LoWPAN). Nous présentons les réseaux 6LoWPANs, leur architecture, leur fonctionnement et leurs domaines d'applications. Enfin, nous soulignons les différents objectifs et défis de sécurité, et les techniques utilisées pour confronter ces défis.

Ce chapitre est organisé comme suit : dans la section 2.1, nous introduisons l'IP-based IoT et ces infrastructures. La section 2.2 est consacrée à la technologie 6LoWPAN, où nous définirons les caractéristiques et les exigences de ce type de réseau. Dans la section 2.3, nous examinons les objectifs et les facettes de sécurité des systèmes IoT. Enfin, nous concluons le chapitre dans la section 2.5.

2.1 L'IoT basé sur IP

L'Internet des objets (IoT) est destiné à connecter tous les objets intelligents qui nous entourent dans notre vie quotidienne. L'IoT vise à assurer des conditions de vie plus intelligentes et une meilleure qualité de service dans différents domaines d'application, tels que : les villes intelligentes, les systèmes de transport intelligents et les soins de santé intelligents [1].

Les réseaux de capteurs sans fil (WSN), qui sont une partie intégrante des applications IoT, nécessitent une certaine interconnexion avec Internet. Les communications de bout en bout entre les dispositifs de détection et d'autres entités Internet sont devenues une exigence fondamentale pour de nombreuses applications de détection. Cette connexion permet l'exploitation des rapports de détection, qui sont souvent massifs, par des applications et services web. La connexion des WSN à Internet nécessite l'adaptation des technologies et d'infrastructures IP aux contraintes des WSNs. Ceci construit une nouvelle technologie sous-jacente pour l'IoT, à proprement parler l'IoT basé sur IP (IP-based IoT). Cette technologie permet aux

capteurs de communiquer et d'échanger des données directement avec tout équipement connecté au réseau IP. Ceci permet également la création des réseaux personnels sans fil à faible puissance qui sont directement connectés à Internet. Cependant, ces réseaux sont connus par leurs contraintes de consommation d'énergie et de faible puissance. Par conséquent, les communications dans des tels réseaux, doivent être prises en charge par un standard qui définit le fonctionnement de leurs appareils et qui prend en charge leurs contraintes.

2.2.1. Le standard IEEE 802.15.4

Les périphériques des WSN sont caractérisés par leurs faibles capacités de calcul et leurs batteries et mémoire limitées, cela conduit à l'introduction du standard IEEE 802.15.4 [2]. Ce standard définit le fonctionnement des réseaux personnels sans fil à faible débit (LoWPAN) au niveau de la couche physique (PHY) et la couche de contrôle d'accès moyen (MAC). Le standard IEEE 802.15.4 prend en charge les communications à 250 Kbit/s dans une courte portée d'environ 10 mètres. La longueur d'une trame IEEE 802.15.4 au niveau de la couche physique est de 127 octets. La trame inclut au maximum l'en-tête de 25 octets de la sous-couche MAC et 102 octets restent disponibles pour la charge utile IP. De ces 102 octets, les en-têtes IP suivis par les extensions facultatives et l'en-tête UDP prennent environ 48 octets. Donc, il ne reste que 54 octets pour la charge utile UDP au niveau de la couche application. Il est évident qu'une adaptation doit être introduite pour prendre en charge l'unité de transmission maximale du protocole IPv6, qui est égale à 1280 octets.

Le protocole IPv6 est considéré comme l'une des technologies candidate pour IoT. L'utilisation des paquets IPv6 aide au déploiement d'un nombre important de dispositifs (objets). Le protocole IPv6 satisfait les performances d'évolutivité, puisqu'il offre un espace d'adressage quasi-illimité. L'adressage IPv6 augmente la taille de l'adresse IP de 32 bits à 128 bits (2^{128} adresses uniques). Par conséquent, pour permettre la communication inter-couche, un paquet IPv6 doit être fragmenté en plusieurs trames 802.15.4 (Voir Figure 2.1), et ensuite toutes les trames 802.15.4 reçues doivent être rassemblée pour régénérer le paquet IPv6 d'origine. Ceci est fait au niveau de la couche adaptation.

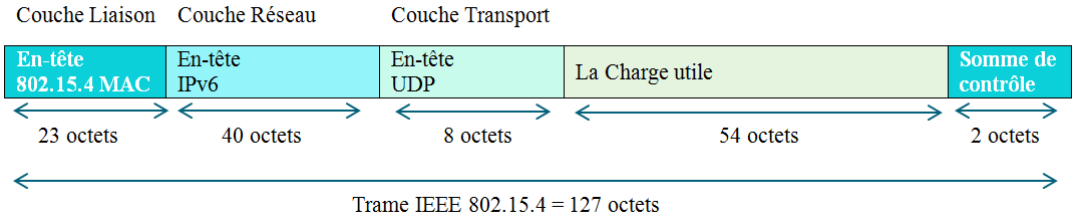

Figure 2. 1 La trame IEEE 802.15.4

2.2.2. Le protocole 6loWPAN

L'Internet Engineering Task Force (IETF) a créé le groupe de travail qui permet l'adaptation du protocole IPv6 pour les réseaux personnels sans fil à faible puissance (6LoWPAN) [3]. Le protocole 6LoWPAN définit les configurations nécessaires et la manière de transporter les datagrammes IP sur des liaisons IEEE 802.15.4. Ce protocole permet aux appareils avec une faible puissance et des capacités de traitement limitées, de participer à l'Internet des objets. Cette possibilité est garantie grâce à un ensemble de modifications et de configurations effectuées au niveau de la couche adaptation de la pile du protocole 6LoWPAN. Plus de détails sur le fonctionnement de cette couche sont présentés dans la section suivante.

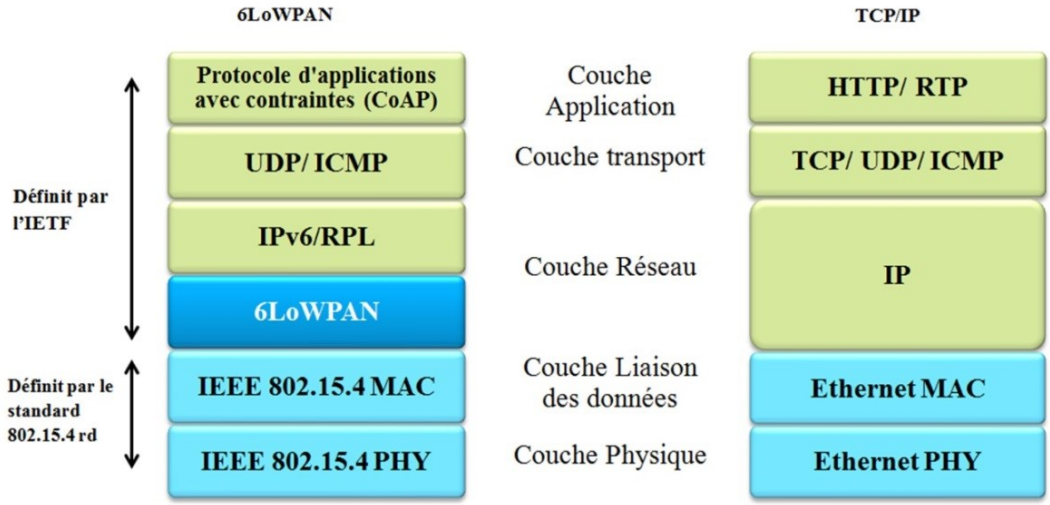

Figure 2. 2 La pile du 6LoWPAN Vs TCP/IP

6LoWPAN conçoit une nouvelle couche d'adaptation ajoutée au modèle OSI, placée au-dessus de la sous-couche MAC IEEE 802.15.4 (Voir Figure 2.2). Cette couche remplit les opérations de compression d'en-tête IPv6, afin de garantir que chaque

périphérique est compatible avec le protocole IPv6. Les opérations de compression permettent d'obtenir une faible surcharge et permettent de transmettre les datagrammes du protocole IPv6 dans des trames IEEE 802.15.4.

Les WSNs basés sur la technologie 6LoWPAN, noté WSNs IP-connectés dans le reste de la thèse, peuvent être étroitement intégrés aux infrastructures IP existantes. L'utilisation d'adressage IPv6 dans les 6LoWPANs présente plusieurs avantages, tels que la possibilité d'utiliser les technologies IP déjà existantes et approuvées. De plus, la connexion des objets connectés IP à d'autres réseaux IP ne nécessitera pas d'entités intermédiaires, comme les passerelles. On peut dire que la technologie 6LoWPAN a changé une perception antérieure de l'IPv6 comme étant peu pratique pour les environnements de communication sans fil. Par conséquent, l'essor de l'IoT est rendu possible grâce à cette technologie qui exploite l'Internet comme une infrastructure de support pour les réseaux de capteurs.

2.2 Les réseaux 6LoWPANs

Un réseau 6LoWPAN est composé d'un nombre de nœuds de capteurs intelligents qui peuvent utiliser IPv6 pour communiquer avec Internet. L'intégration de réseaux personnels sans fil à faible puissance à Internet permet à un grand nombre d'objets intelligents, de récolter les données et les informations.

2.2.1. Architecture du réseau

L'architecture 6LoWPAN est composée de réseaux sans fil de faible puissance, qui sont des réseaux IPv6 stub[1]. Donc, c'est une collection des nœuds qui partagent un préfixe d'adresse IPv6 commun (les 64 premiers bits d'une adresse IPv6), ce qui signifie que peu importe où se trouve un nœud dans un réseau 6LoWPAN, son adresse IPv6 reste la même. Un réseau 6LoWPAN peut être : un réseau simple, un 6LoWPAN ad hoc qui fonctionne sans infrastructure, ou bien un réseau étendu qui englobe plusieurs routeurs de bordure interconnectés avec une liaison dorsale (Ethernet).

[1] Un réseau stub est un réseau auquel les paquets IP sont envoyés ou destinés, mais qui n'agit pas comme un transit vers d'autres réseaux. Ce type de réseau ne connaît qu'une route par défaut vers des destinations non locales. Il envoie son trafic non local via un seul chemin.

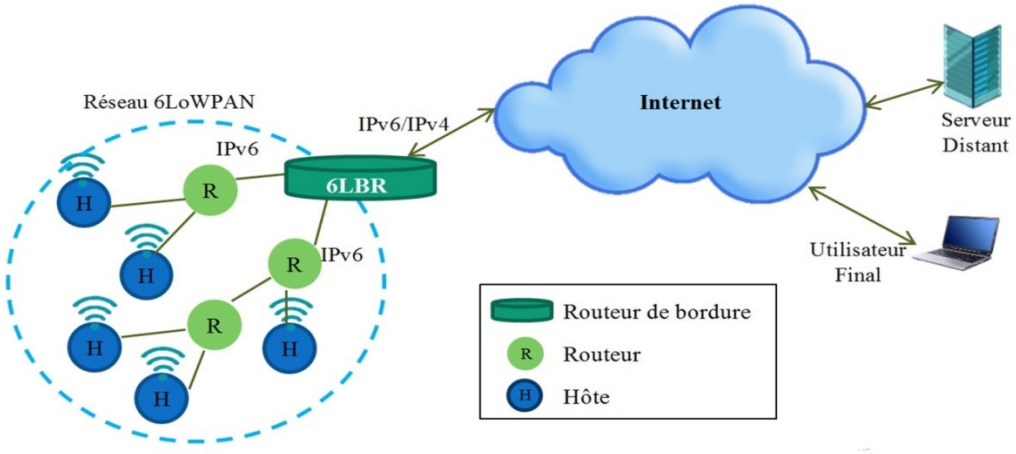

Figure 2. 3 Architecture d'un réseau 6LoWPAN

La figure 2.3 présente une architecture 6LoWPAN simple. Il est donc possible de distinguer trois acteurs dans un réseau 6LoWPAN :

- **Le routeur de bordure (6LBR) :** C'est la racine d'un réseau 6LoWPAN qui le connecte à un autre réseau IP. Le routeur de bordure est un équipement ayant la totalité des fonctions, en anglais Full Function Device (FFD)[2]. Cet équipement joue un rôle important dans l'acheminement du trafic 6LoWPAN entrant et sortant. Il est responsable de la compression et de la découverte des voisins. En outre, dans le cas où un réseau 6LoWPAN doit être connecté à un réseau IPv4, le routeur de bordure gérera également l'inter-connectivité IPv4.
- **Le routeur (6LR) :** Equipement intermédiaire ayant la totalité des fonctions (FFD), dont le but est d'étendre la superficie du réseau.
- **L'hôte (H):** C'est l'équipement feuille du réseau, ayant des fonctions réduites (RFD). Il s'agit des dispositifs extrêmement simples avec des besoins très modestes en ressources et en communication. Ils ne peuvent communiquer qu'avec les FFD et ne peuvent jamais agir en tant que coordinateurs.

2.2.2. La couche adaptation

Les 6LoWPANs permettent une utilisation efficace de l'IPv6 sur des réseaux sans fil à faible puissance via la couche d'adaptation. Cette couche est responsable de

[2] Equipement ayant la totalité des fonctions: défini par IEEE 802.15.4 comme un nœud ayant des niveaux de fonctionnalité complets. Il peut être utilisé pour envoyer et recevoir des données, mais il peut également acheminer des données à partir d'autres nœuds.

plusieurs opérations : la compression d'entête IP, la fragmentation et le réassemblage des paquets, l'adressage sans état, le routage Mech-under[3] et la cohérence avec les couches supérieures.

2.2.2.1. Compression des entêtes

La couche d'adaptation permet la compression des entêtes d'IPv6 et d'entêtes suivants tels qu'UDP. La compression d'en-tête comprime les 40 octets de l'en-tête IPv6 et les 8 octets de l'en-tête UDP. La compression se base sur l'utilisation des champs communs, et en se basant sur l'espace d'adressage hiérarchique d'IPv6 qui permet la plupart du temps, d'éluder complètement les adresses IPv6.

Les en-têtes 6LoWPAN sont définis dans [4], et sont améliorés et étendus tardivement dans [5]. L'en-tête se compose d'une valeur de répartition (Dispatch) identifiant le type d'en-tête, suivie d'un octet de compression d'en-tête IPv6 indiquant les champs compressés, puis les champs IPv6 en ligne. Si, par exemple, les en-têtes d'extension suivent IPv6, ces en-têtes peuvent également être compressés à l'aide de ce qu'on appelle la compression de l'en-tête suivant [5]. La couche d'adaptation supprime les informations dupliquées qui peuvent être dérivées d'autres couches, telles que les adresses IPv6 et les champs « Longueur ». Donc, 6LoWPAN utilise deux types de compression d'en-tête: la compression d'en-tête IPv6 LOWPAN_IPHC et la compression d'en-tête UDP LOWPAN_NHC**.** Dans le cas de LOWPAN_IPHC, les champs d'en-tête suivants sont candidats à la compression: version, identificateurs d'interface IPv6, longueur de paquet, classe de trafic, étiquette de flux et le type d'en-tête suivant. Le seul champ qui ne peut pas être compressé est « limite de saut ». Cependant, dans le cas de LOWPAN_NHC, les champs suivants peuvent être compressés: port source, port de destination et longueur. La somme de contrôle de l'en-tête UDP est le seul champ qui ne puisse pas être compressé. Un exemple de compression d'entêtes IPv6 et d'UDP ainsi que LOWPAN_IPHC et LOWPAN_NHC sont présentés dans la figure 2.4. LOWPAN_IPHC réduit la longueur de l'en-tête IP à 2 octets pour un réseau à un seul saut et à 7 octets dans le cas d'un réseau à sauts multiples. LOWPAN_NHC a une longueur de 1 octet ou

[3] Lorsque le routage des paquets est effectué par la couche réseau, il est appelé Route-over. Cependant, lorsque le routage est mis en œuvre au niveau de la couche d'adaptation, il est appelé Mesh-under.

plus ; les premiers bits de longueur variable identifient le type d'en-tête suivant et les bits restants sont utilisés pour coder les informations d'en-tête.

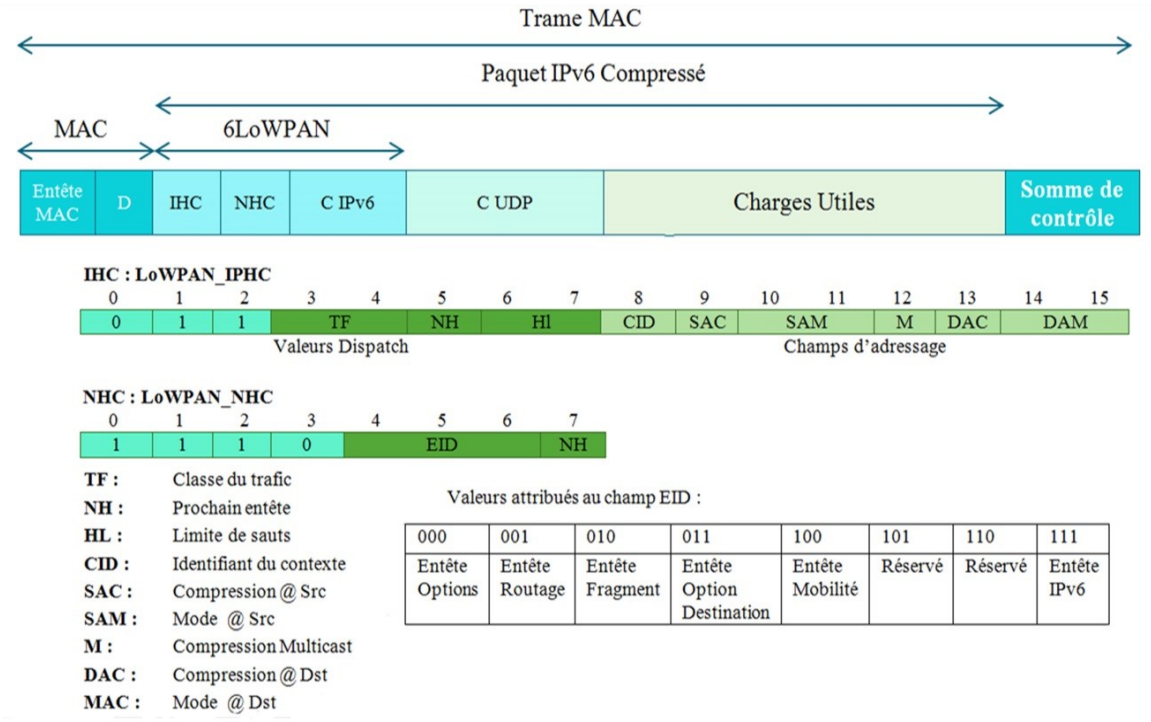

Figure 2. 4 Compression 6LoWPAN

2.2.2.2. *Fragmentation des paquets*

Lorsque les données dépassent la longueur d'une seule trame 802.15.4, 6LoWPAN effectue la fragmentation. Pour cette raison, les trames IPv6 doivent être divisées en plusieurs petits segments. Cette opération est appelée la fragmentation des paquets.

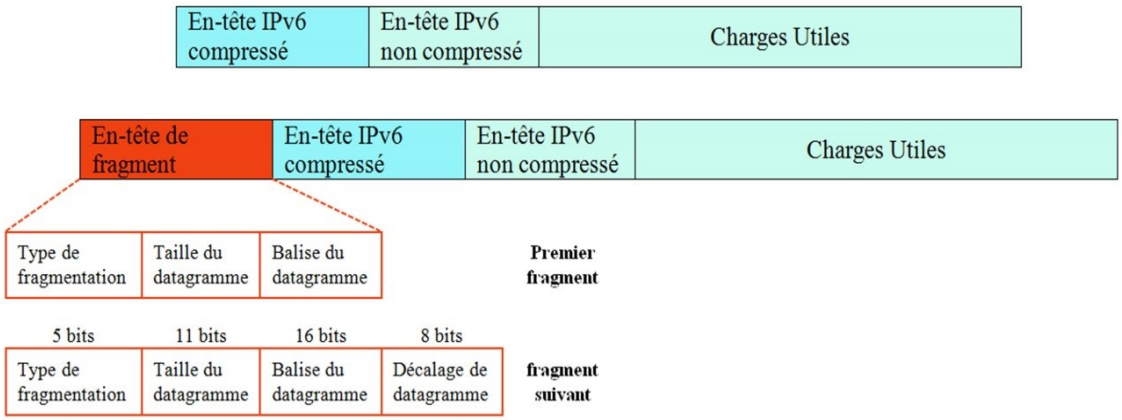

Figure 2. 5 Fragmentation 6LoWPAN

Lorsque la fragmentation est utilisée, un en-tête de fragment est ajouté au paquet. L'en-tête du fragment contient trois champs : taille du datagramme, balise de datagramme et décalage du datagramme. La taille du datagramme décrit la charge utile totale (non fragmentée). La balise de datagramme identifie l'ensemble de fragments, utilisée pour faire correspondre des fragments de la même charge utile. Le décalage de datagramme identifie le décalage du fragment dans la charge utile non fragmentée (Voir figure 2.5). La longueur de l'en-tête de fragment est de 4 octets pour le premier en-tête et de 5 octets pour tous les en-têtes suivants. Lorsque les paquets de données sont réassemblés, les informations supplémentaires ajoutées sont supprimées et les paquets sont restaurés à leur format IPv6 initial. La séquence de fragmentation est différente selon le type de routage utilisé.

Dans le cas du routage Mech-under, les fragments sont réassemblés uniquement à leur destination finale, tandis que dans le cas des réseaux Route-over, les paquets de données sont réassemblés à chaque saut. Ainsi, dans un réseau avec routeur, chaque tronçon doit avoir suffisamment de ressources pour stocker tous les fragments. Alors que dans un système maillé, beaucoup de trafic réseau est généré rapidement, car tous les fragments sont transmis immédiatement. Par conséquent, la fragmentation doit être évitée au maximum, car elle a un impact négatif sur la durée de vie des batteries qui alimentent les appareils 6LoWPAN.

2.2.3. Applications 6LoWPAN

Les applications basées WSN peuvent aller de la surveillance de la santé personnelle à la surveillance des installations à grande échelle, ce qui diffère considérablement. Cela contraste avec la technologie de l'information sur PC, qui est assez homogène et vise principalement les environnements domestiques et professionnels. L'utilisation idéale de 6LoWPAN est optimale dans des applications avec des appareils embarqués qui doivent communiquer avec des services Internet, ou bien pour relier des réseaux hétérogènes de faible puissance, où le réseau doit être ouvert, réutilisable et évolutif pour de nouveaux usages et services. En gros, la connexion d'Internet au monde physique permet un large éventail d'applications intéressantes. La technologie 6LoWPAN peut être applicable, par exemple:

- La domotique et l'automatisation de santé, automobile et d'industrie.

- Les infrastructures de comptage intelligent (smart metering) et les réseaux intelligents (smart Grid).
- La surveillance et les prévisions environnementales en temps réel.
- Les systèmes de sécurité et de défense.
- Dans la transmission de données événementielle.

Malgré que 6LoWPAN est efficace pour plusieurs applications internet, nous devons prendre en considération les contraintes liés à ce type de réseaux, tels que:

- La taille minimale des paquets 6LoWPANs, le protocole de routage doit imposer une surcharge minime (voire aucune) aux paquets de données, indépendamment du nombre de sauts.
- la nature des dispositifs 6LoWPAN caractérisé par des ressources d'énergie, de traitement et de stockage limitées, en plus des liaisons sans fil avec perte, les déploiements non surveillés et la communication multi-sauts.
- Les opérations de calcul et l'utilisation de mémoire doivent être minimales pour répondre aux objectifs de faible coût et de faible consommation.
- La non-fiabilité des périphériques 6LoWPAN liée à la connectivité radio incertaine, décharge de la batterie, blocages de périphériques, altération physique, etc.
- Fournir des services de sécurité dans 6LoWPAN, car les appareils sont extrêmement hétérogènes, principalement déployés dans des environnements sans surveillance et plus proches des humains que les nœuds WSN typiques.

2.3 Sécurité des réseaux 6LoWPAN dans le contexte de L'IoT

Les WSNs, par leur nature, sont largement ouverts aux attaquants, qui peuvent entendre et injecter les paquets échangés. Le problème de sécurité dans les WSNs IP-connectés est un sujet controversé dans l'IoT. Dans l'enquête menée par Nguyen et al. [6], les auteurs ont souligné que les protocoles de sécurité Internet légers sont davantage recommandés pour réduire la complexité de la communication. Une revue plus récente est présentée dans [7], où les auteurs ont proposé une classification des différents mécanismes utilisés pour assurer la sécurité de bout en bout dans l'IoT. En

effet, un système ne peut être qualifié, sécurisé que si les objectifs de sécurité prédéfinis sont tous atteints.

2.3.1. Objectifs de sécurité

IPv6 offre l'interconnexion de la plupart des objets physiques avec Internet. Cela ouvre plus de possibilités pour développer des nouvelles applications pour l'IoT. Cependant, la sécurité des déploiements réels présente un véritable défi. Les objectifs de sécurité suivants sont nécessaires dans l'IoT [8].

- **Confidentialité** : les messages qui transitent entre une source et une destination peuvent être facilement interceptés par un attaquant et le contenu secret sera révélé. À cet effet, ces messages doivent être cachés aux entités intermédiaires et non autorisé. Ceci est garanti par la confidentialité des messages de bout en bout (E2E) qui est fortement requise dans l'IoT. Cet objectif peut être garanti grâce aux opérations de chiffrement et de déchiffrement.
- **Intégrité des données** : aucun intermédiaire entre une source et une destination ne devrait pouvoir modifier le contenu secret des messages ni des données stockées. Les codes d'intégrité des messages (MIC) sont principalement utilisés pour garantir cet objectif.
- **Authentification** : les points terminaux communicants devraient pouvoir vérifier les identités des uns et des autres. Cette vérification garantie qu'ils communiquent avec les entités qu'ils prétendent être. Différents schémas d'authentification sont utilisés pour garantir cet objectif.
- **Disponibilité** : pour le bon fonctionnement de l'IoT, il est également important que les services proposés par les applications fonctionnent correctement et constamment. En d'autres termes, les intrusions et les activités malveillantes doivent être détectées. Les systèmes de détection d'intrusion (IDS) et les pare-feu (firewalls), en plus de divers mécanismes de sécurité, sont utilisés pour garantir cet objectif.
- **Protection Anti Rejeu** : un nœud intermédiaire compromis peut stocker un paquet de données et le rejouer ultérieurement. Le paquet rejoué peut contenir des données de détection typiques, comme il peut contenir une demande de service payant. Il est donc important qu'il y ait des mécanismes pour détecter

les messages rejoués ou en double. Cet objectif est réalisé grâce à des timetamps, des numéros de séquence et des nonces protégés par intégrité.

2.3.2. Facettes de sécurité

Dans le but de fournir une sécurité à multiples facettes, nous devons assurer la sécurité des réseaux 6LoWPAN, la sécurité des périphériques, c'est-à-dire la sécurité des données secrètes stockées, ainsi que la sécurité des communications E2E. La Figure 2.6 présente une hiérarchie des trois grandes facettes de sécurité dans l'IoT.

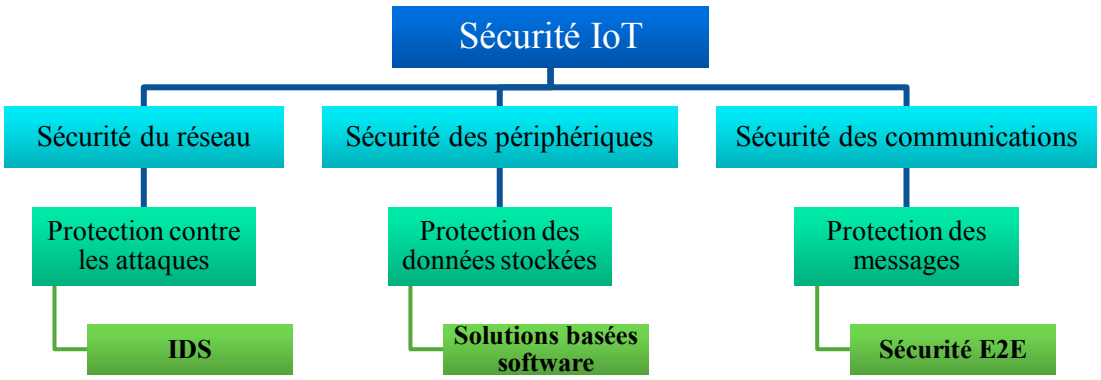

Figure 2. 6 Facettes de sécurité d'IoT

La sécurisation d'un réseau contre les attaques malveillantes est primordiale. Diverses attaques contre l'IoT et les WSN ont été identifiées dans [9]. Le but de ces attaques est de perturber le fonctionnement des réseaux en interrompant la topologie de routage ou en lançant des attaques dénis de service (DoS). Ceci peut être limité par l'utilisation des systèmes de détection d'intrusion (IDS), qui sont nécessaires pour détecter les imposteurs et les activités malveillantes dans le réseau. En outre, les pare-feu sont également nécessaires pour bloquer l'accès non autorisé aux réseaux.

Dans l'IoT, les réseaux 6LoWPAN sont vulnérables à un certain nombre d'attaques provenant d'Internet et de l'intérieur du réseau. Les nœuds 6LoWPAN sont relativement plus faciles à compromettre que les nœuds classiques. Ce qui fait qu'réseau 6LoWPAN peut lui-même devenir une source d'attaques contre les hôtes Internet. Plusieurs recherches ont été menées dans ce domaine. Par conséquent, plusieurs IDS ont été proposés pour les WSNs IP-connecté. L'objectif principal de ces IDS est de détecter les attaques courantes sur les WSN, telles que les attaques

trou de creux[4] (Sink-Hole) [10] et trou de ver[5] (Worm-Hole) [11]. D'autres chercheurs ont utilisé différentes technique pour développer leurs IDS comme dans [12], ou les auteurs ont regroupé le réseau en petits groupes appelés clusters. L'instance IDS proposée est placée dans chaque tête de cluster qui surveille les membres du cluster en reniflant leurs communications.

En plus de la sécurisation du réseau contre les attaques, il est important de protéger les données sensibles stockées dans un appareil IoT. La plupart des appareils IoT sont des nœuds minuscules à ressources limitées. Dans un modèle de stockage typique, les données sont stockées sous une forme cryptée avec leur hachage cryptographique. Lorsqu'un hôte distant demande des données, elles sont décryptées, rechiffrée, protégée par l'intégrité et puis transmises. Il existe des solutions logicielles diverses qui peuvent être utilisées pour sécuriser les données de l'IoT IP-connectés cryptographiquement. Les solutions les plus rentables sont présentées dans [13-14].

La sécurité des communications est le domaine sur lequel on a focalisé dans ce travail. La communication dans l'IoT doit être protégée tout en garantissant les objectifs de sécurité abordés dans la section précédente. Il est important que les messages échangés soient protégés en confidentialité et en intégrité. Les réseaux 6LoWPAN utilisent le protocole IEEE 802.15.4 au niveau de la couche de liaison. Cette couche protège une communication sur une base par bond à l'aide de la sécurité offerte par la norme 802.15.4 [15], où chaque nœud du chemin de communication doit être approuvé. Cependant, cette norme prend uniquement en charge la sécurité saut par saut (hop by hop). Ce mécanisme est difficile à mettre en œuvre sur des nœuds de capteurs à ressources limitées, car ils sont coûteux en termes de taille de code et de vitesse de traitement. De plus, les messages quittant le réseau 6LoWPAN et continuant à circuler sur un réseau IP ne sont pas protégés par les mécanismes de sécurité de la couche liaison. Par conséquent, les données circulant entre les hôtes

[4] Une attaque Sink-Hole est l'une des attaques de routage les plus sévères car elle attire les nœuds environnants avec des informations de chemin de routage trompeuses et effectue un forage de données ou un transfert sélectif des données qui la traversent.

[5] L'attaque trou de ver nécessite la coopération de deux ou plusieurs nœuds situés à des zones différentes du réseau. Les nœuds malveillants mettent en place un tunnel de communication privé entre eux où les paquets reçus d'un côté sont retransmis de l'autre côté.

Internet et les routeurs de bordure doivent être également sécurisés. En d'autres termes, la sécurité E2E doit être assurée.

2.3.3. La sécurité des communications de bout en bout

La sécurité de bout en bout (E2E) protège la communication tout au long du chemin entre deux nœuds communicants. C'est un élément important de tout système de sécurité robuste, à tel point que cette exigence est devenue une fonctionnalité essentielle dans le développement des 6LoWPANs.

2.3.3.1. Le standard 802.15.4

La sécurité assurée par le standard 802.15.4 au niveau de la couche liaison de donnée [16], est bénéfique en termes de performances. Ceci est dû à la prise en charge matérielle des fonctions cryptographiques par les puces radios 802.15.4. Par contre, ni l'authentification de l'hôte ni la gestion des clés ne sont prises en charge. Les messages quittant le réseau de capteurs et continuant à circuler sur un réseau IP ne sont pas protégés par les mécanismes de sécurité de couche liaison. Ainsi, comme chaque nœud du chemin doit être approuvé, la sécurité E2E dans l'IoT ne peut pas être obtenue à l'aide de cette l'approche.

2.3.3.2. L'IPsec compressé

Dans Internet, l'IoT et l'IP-based IoT, la sécurité au niveau de la couche réseau peut être assurée par le protocole IPsec [17]. Il existe un nombre croissant de travaux qui confirment l'efficacité d'IPsec pour les 6LoWPANs. IPsec est une technologie mature et éprouvée, mais un protocole de sécurité lourd. Par conséquent, IPsec a besoin de certaines modifications pour s'adapter aux environnements avec contraintes.

Une proposition académique est présentée par Raza et al. dans [18-19], pour étendre 6LoWPAN avec IPsec en utilisant des techniques de compression d'en-tête [20]. Selon les auteurs, des capacités de sécurité peuvent être ajoutées à IP en utilisant la compression d'en-tête IPHC et la compression d'en-tête suivante (NHC). Le codage NHC consiste en un octet NHC_EH comprenant trois bits pour l'ID d'en-tête d'extension (EID), soit huit valeurs. Deux emplacements libres « 101 » et « 110 » restent disponibles et seront utilisés pour indiquer qu'un prochain en-tête sera soit pour le protocole Authentication Header (AH) ou bien pour Encapsulating Security Payload (ESP). Dans ce cas, le champ «Next Header» est défini sur 1. Comme

illustré sur la figure 2.7, les quatre premiers bits sont l'ID NHC pour AH, réglé sur «1101». Les champs SPI et SN définissent respectivement le taux de compression de l'index des paramètres de sécurité et le numéro de séquence dans l'en-tête AH. Le champ «Longueur» peut être élidé. La taille ICV peut être dérivée de la valeur SPI car la longueur de la somme de contrôle dépend de l'algorithme cryptographique utilisé. Quant à ESP, Les quatre premiers bits représentent l'ID NHC pour ESP, défini sur « 1110 ». Les champs SPI et SN sont les mêmes que pour AH. La longueur minimale de l'entête ESP sans authentification est de 18 octets avec AES-CBC et un alignement parfait des blocs. Après la compression, l'en-tête ESP peut être réduit à 12 octets. Lorsque l'ESP fournit l'authentification, 12 octets doivent être ajoutés pour l'ICV.

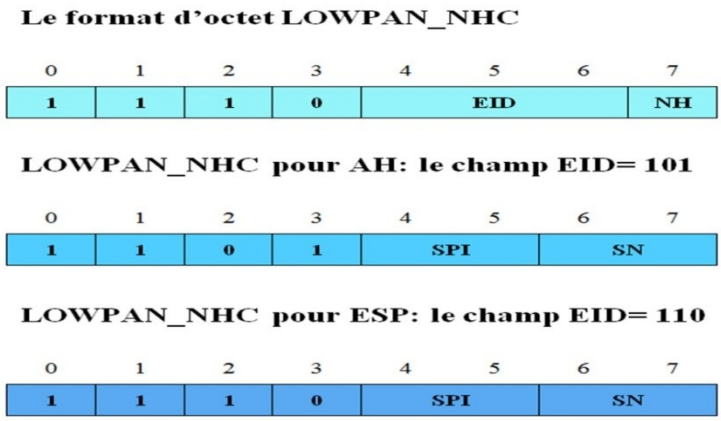

Figure 2. 7 Compression d'entête IPsec

Par ailleurs, il existe d'autres recherches qui ont proposé l'utilisation du protocole d'identité des hôtes (HIP) [21], pour sécuriser les communications E2E dans 6LoWPAN. Par conséquent, différentes version de ce protocole ont été proposées tel que : HIPDEX [22], Slimfit [23] et HIP-TEX [24].

2.3.3.3. Le protocole HIPDEX

Le protocole HIP Diet Exchange (HIP-DEX) a été proposé pour réduire le coût de calcul des échanges de base du protocole HIP. Cette réduction est due à l'utilisation de la cryptographie à courbe elliptique (ECC). Une seule clé publique est nécessaire pour calculer la clé de session. La détention de la clé de session est suffisante pour authentifier le nœud et prouver sa légitimité. Bien que la solution proposée semble

assez simple et légère, la modification des formats de message dans le protocole HIP standard, en supprimant certains champs, peut introduire des problèmes d'incompatibilité lors de l'établissement de la session avec des hôtes Internet utilisant le HIP d'origine. De plus, il offre un niveau de sécurité minimum par rapport au coût de calcul important.

2.3.3.4. Le protocole SLIMfit

Le protocole SLIMfit est un développement de son prédécesseur HIP-DEX. Il s'agit d'une couche HIP-DEX compressée qui utilise un mécanisme de reprise de session. Elle vise à compresser les informations redondantes et à omettre les informations non pertinentes. SLIMfit utilise les courbes elliptiques avec Diffie-Hellman pour l'échange de clés dans des scénarios non collaboratifs. Cette solution peut convenir à l'IoT car elle présente les avantages de la résilience, de la mémoire et des compétences de communication. Cependant, elle ne fournit pas l'évolutivité et l'interopérabilité.

2.3.3.5. Le protocole HIPTEX

Le protocole HIP Tiny Exchange (HIPTEX), est un protocole d'échange de clés léger distribué. Malgré le fait que HIP-TEX ait un faible coût de calcul en raison de l'élimination des échanges Diffie-Hellman, le protocole est pauvre en termes de mécanismes de sécurité comme l'authentification mutuelle et l'échange de clés. En outre, bien que HIP-TEX soit relativement efficace en termes de calcul et de mémoire, il manque d'efficacité de communication, car HIP-TEX entraînera plus de trafic IoT [25].

2.3.3.6. Le protocole 6LowPSec

Les auteurs ont introduit un protocole de sécurité de bout en bout qui fonctionne au niveau de la couche Adaptation, nommée 6LowPSec [26]. Ce protocole bénéficie à la fois d'IPsec et des fonctionnalités de sécurité matérielle existantes de la couche MAC IEEE 802.15.4. Puisque il fonctionne au niveau de la couche adaptation, il nécessite le minimum des charges, de traitement et d'échange d'information de contrôle. Il exécute des fonctions de sécurité (chiffrement, vérification d'intégrité, authentification) uniquement sur les terminaux. Donc, un routeur 6LoWPAN de

frontière ne nécessite pas des fonctions de sécurité supplémentaires. Cependant, 6LowPSec définit une méthode de gestion de clés basique et non développée.

2.3.3.7. Le protocole TLS

La sécurité E2E peut être également assurée par le protocole de sécurité de la couche transport (TLS). En opérant entre la couche transport et la couche application, TLS garantit la sécurité entre les applications. TLS comprend un mécanisme d'échange de clés et fournit une authentification entre les hôtes internet en plus de la confidentialité et l'intégrité. En contrepartie, TLS ne peut pas être utilisé que sur le protocole de contrôle de transmission (TCP). Or, ce protocole, n'est pas la méthode de communication préférée pour les objets intelligents IP-connectés; en raison de leurs faibles puissances et la difficulté de maintenir une connexion continue dans les réseaux 6LoWPANs.

2.3.3.8. Le protocole DTLS

Le protocole DTLS est une adaptation de TLS pour UDP appelée Datagram TLS (DTLS) [27]. Il garantit la sécurité E2E des différentes applications sur une même machine, il opère entre les couches transport et application. En plus des fonctionnalités assurées par TLS, DTLS offre une protection contre les attaques par déni de service (DoS) grâce à l'utilisation de cookies. Bien que DTLS offre une sécurité E2E au niveau applicatif, il ne peut être utilisé que via le protocole UDP. Pour que DTLS soit faisable pour l'IoT, il doit subir des adaptations pour convenir aux réseaux 6LoWPANs. DTLS est un protocole lourd et ses en-têtes sont trop longs pour tenir dans un seul MTU IEEE 802.15.4. Donc, afin d'alléger le protocole et réduire le nombre de bits de sécurité supplémentaires, une compression d'en-tête pour DTLS est nécessaire [28].

Kothmayr et al. [29] étudient l'utilisation de DTLS pour les réseaux 6LoWPAN avec un module de plateforme sécurisée (TPM), afin d'obtenir la prise en charge matérielle de l'algorithme RSA. Cependant, ils ont utilisé DTLS tel quel il est. Aucune de méthode de compression n'est utilisée pour raccourcir les bits redondants dans les messages DTLS. Par ailleurs une modification du protocole DTLS est effectuée dans [30]. Les auteurs ont proposé une architecture de sécurité 6LoWPAN en fonction du routeur de bordure (6LRB). Leur architecture prend en charge les

communications de couche transport de bout en bout et garantit une authentification mutuelle à l'aide de la cryptographie à clé publique. Une autre implémentation du protocole DTLS a été réalisée dans [31], où les auteurs ont proposé une implémentation d'un DTLS compressé avec un mode de clé publique brute. L'objectif principal de cette mise en œuvre est l'établissement de communications sécurisées entre les appareils via le protocole CoAP. Il est à noter que cette mise en œuvre minimise la consommation d'énergie. En ce qui concerne l'aspect des performances du protocole DTLS, une amélioration de la sécurité E2E associée à une réduction du cout de calcul est proposée dans [32]. Les auteurs considèrent la sécurité E2E orientée groupe. Chaque client « CUP » a l'autorisation d'accéder à plusieurs serveurs CoAP dans un groupe pré affecté.

2.3.3.9. Le protocole CoAPs

Dans les systèmes IoT, la version sécurisée du protocole d'applications avec contraintes (CoAPs) [33], rend l'utilisation de DTLS obligatoire comme solution de sécurité sous-jacente pour CoAP [34]. Une proposition du protocole CoAP pour soutenir la sécurité est présentée dans [35]. L'approche proposée est basée sur trois nouvelles options. La première permet d'identifier la façon dont la sécurité est appliquée à un message CoAP donné, et permet aussi d'identifie l'entité responsable du traitement de la sécurité pour ce message. La deuxième option permette le transport des données nécessaires pour l'authentification et l'autorisation d'un client CoAP. La troisième option permet le transport des données nécessaires au traitement de la cryptographie pour un message CoAP. Cette approche permet une sécurité granulaire par message, et prend en charge l'utilisation de plusieurs mécanismes d'authentification.

2.3.4. Etude comparatives des protocoles de sécurité.

Plusieurs protocoles ont été proposés pour assurer la sécurité des communications dans les réseaux 6LoWPAN. La sécurité peut être fournie à différents niveaux de la pile 6LoWPAN. Le tableau 2.4 présente une comparaison entre ces travaux. Chaque proposition est caractérisée par sa couche ciblée, son mécanisme proposé pour assurer la sécurité, ses avantages et ses inconvénients.

Ref	Couche 6LoWPAN	Mécanisme proposé	Avantages	Inconvinients
[16]	Couche de Laison des données	IEEE 802.15.4	⊕ Assure la sécurité saut par saut. ⊕ Prise en charge matérielle des fonctions cryptographiques.	∅ l'authentification de l'hôte non assurée. ∅ Absence de mécanisme de gestion des clés.
[26]	Couche Adaptation	6LowPSec Protocol	⊕ Garantit le sécurité E2E au niveau de la couche adaptation. ⊕ Bénéficie de l'assistance matérielle fournis par le standard 802.15.4. ⊕ Combiné avec IPsec	∅ Définit un mécanisme de gestion de clé basique et non développé
[18] [19]	Couche Réseau	Compressed IPsec	⊕ Inclus un mécanisme de gestion de clé (IKEv2). ⊕ Interoperable avec les protocoles TCP. ⊕ Assure la confidentialité et l'intégrité pour la couche Transport.	∅ Une charge et une complexité de calcul causé par IKEv2. ∅ Les failles de sécurité relatives à IPsec et à IKEv2.
[23]	Couche Réseau	SLIMfit (Version du HIP)	⊕ la résilience de la mémoire. ⊕ Forte compétences de communication.	∅ Ne fournit pas l'évolutivité et l'interopérabilité
[24]	Couche Réseau	HIPTEX	⊕ Efficace en termes de calcul et de mémoire. ⊕ faible coût de calcul	∅ pauvre en termes de mécanismes de sécurité (l'authentification mutuelle et l'échange de clés)
[25]	Couche Transport	TLS	⊕ Inclus un mécanisme de gestion des clés. ⊕ Assure l'authentification des hotes.	∅ Nécessite TCP comme protocole de transport sous-jacent
[29]	Transport/ Application	DTLS	⊕ Utilise l'assistance matérielles pour éxécuter les opérations crypttographique. ⊕ Garantit une authentification mutuelle	∅ La prise de contact DTLS ne convient pas aux périphériques contraints. ∅ Impossible de protéger les en-têtes IP, UDP et ICMP. ∅ Il est étroitement intégré dans les WSN basés sur IP
[30]	Transport/ Application	DTLS Compréssé	⊕ Utilise une version compressée du protocole DTLS. ⊕ Réduit la consommation d'énergie	
[31]	Transport/ Application	DTLS	⊕ Réduit de la charge de calcul	
[35]	Application	CoAP	⊕ Assure une sécurité transparente de bout en bout de la couche application	∅ Hérite des inconvénients de DTLS car il l'utilise comme protocole de sécurité

Tableau 2. 1 Protocoles proposés pour la sécurité d'IoT IP-Connecté

En ce qui concerne la sécurité E2E, elle peut être assurée au niveau de la couche réseau, en utilisant le protocole HIP ou IPsec. Malgré ses divers avantages, le protocole HIP ne peut pas être adopté pour les 6LoWPAN tel qu'il est. La communication des longs messages et le calcul des opérations cryptographiques asymétriques coûteuses, présentent les principales sources de consommation élevée d'énergie, de surcharge de mémoire et d'épuisement des ressources de traitement.

Grâce aux mécanismes de compression d'en-tête IPsec, ce protocole peut convenir aux environnements contraints comme les 6LoWPANs. En effet, nous assumons qu'il est avantageux d'utiliser IPsec, car les points terminaux existants sur Internet n'auront pas besoin d'être modifiés pour communiquer en toute sécurité avec le WSN. De plus, en utilisant IPsec, une véritable sécurité E2E est implémentée et le besoin d'une passerelle de confiance est supprimé. En outre, en utilisant IPsec, nous n'avons pas besoin d'utiliser les mécanismes de sécurité de couche liaison 802.15.4 existants, ce qui à son tour libère de l'espace d'en-tête.

La sécurité de la couche transport peut être assurée par le protocole TCP. Cependant, TLS ne peut être utilisé que sur TCP, qui est rarement utilisé dans les WSNs. Il ne peut pas garantir la confidentialité et l'intégrité des en-têtes des couches supérieures. En outre, TLS et tout autre protocole qui gère la fiabilité des communications et la livraison de paquets en séquence, ne convient pas aux environnements peu fiables et très contraints, tout comme les WSNs.

Malgré les recherches approfondies qui ont été effectuées sur le protocole DTLS, son mécanisme d'authentification et de gestion de clé, basée sur ECC, ne convient pas aux appareils contraints. Ainsi, la fragmentation des messages implique la retransmission et la réorganisation des messages. En outre, DTLS ne peut pas protéger les en-têtes IP, UDP et ICMP, et il est étroitement intégré dans les WSN IP-connectés.

IPsec, étant une solution standard pour Internet, assure la confidentialité, l'intégrité et l'authenticité, et ainsi couvre ces trois exigences de sécurité. Par conséquent, nous assumons dans ce travail qu'IPsec est une solution de sécurité efficace pour les WSNs IP-Connectés. Donc, IPsec peut être utilisé pour sécuriser les communications E2E dans les réseaux 6LoWPANs. Néanmoins, reste à trouver un mécanisme

efficace de gestion de clé qui répond aux exigences des dispositifs IoT. Plus de détails sur le protocole IPsec sont présentés dans le chapitre suivant.

2.3.5. Le Clustering pour la gestion des clés IoT

La technique de clustering consiste à regrouper un ensemble d'objets, tel qu'un ensemble de capteurs dans un groupe nommé un cluster. Chaque cluster a un nœud principale qui préside le groupe nommée tète de cluster (cluster Head). Les objets du même groupe ou du les membres du cluster sont similaires les uns aux autres, et ils sont plus similaires en tant qu'un groupe par rapport aux objets des autres groupes (clusters). L'espace de développement IoT a attiré l'attention des chercheurs pour utiliser le clustering dans les schémas de gestion des clés. Les têtes de cluster sont supposées être dotées de capacités de traitement élevées et déployées dans une topologie de grille. Ainsi, le clustering peut être utilisé pour la création, la distribution et la révocation de clés. Par conséquent, protéger un réseau dynamique contre diverses attaques en préservant ces performances.

2.4 Conclusion

Dans ce chapitre, nous avons présenté le paradigme de l'Internet des Objets basé sur IP. Nous avons présenté le fonctionnement de cette technologie ainsi que ces infrastructures relatives. En particulier, nous avons concentré sur 6LoWPAN qui est la technologie sous-jacente d'IP-based IoT. Nous avons examiné les caractéristiques et les domaines d'application de cette technologie ainsi que l'architecture et les contraintes du réseau. Ensuite, nous avons présenté un aperçu sur l'aspect de sécurité dans l'IoT. Nous avons introduit les solutions utilisées pour chaque domaine de sécurité et particulièrement des communications de bout en bout. Finalement, nous avons effectué une comparaison entre les différents protocoles proposés pour garantir la sécurité dans les réseaux 6LoWPAN. D'où, nous avons démontré que la compression du protocole IPsec, rend ce dernier est une solution efficace pour ce type d'environnement. Dans le chapitre suivant nous allons donner plus de détails sur le protocole IPsec et son fonctionnement.

CHAPITRE III

La sécurité du protocole Internet : IPsec

Dans ce chapitre, nous présentons la sécurité du protocole internet, en anglais Internet Protocol Security (IPsec). En premier lieu, nous introduisons les services de sécurité offerts par ce protocole. Ensuite, nous définirons ses modes de fonctionnement et ses protocoles sous-jacents. Nous présentons son architecture de sécurité en définissant ses éléments essentiels qui assurent les différents services de sécurité. En deuxième lieu, nous présentons une étude sur les travaux proposés dans la littérature pour la vérification des politiques de sécurité, qui est un élément de base d'IPsec. Enfin, nous présentons le protocole IKE utilisé par IPsec pour la gestion des clés et des associations de sécurité.

Ce chapitre est organisé comme suit : dans la section 3.1 nous définirons le protocole IPsec et nous présentons les différents services de sécurité assurés par ce protocole. La section 3.2 définit les deux modes de fonctionnement du protocole. Dans la section 3.3, nous définirons les protocoles sous-jacents d'IPsec ainsi que leurs caractéristiques. Dans la section 3.4, nous nous penchons sur l'architecture de sécurité. Nous examinons dans cette section, les éléments essentiels qui garantissent le bon fonctionnement du protocole. En outre, la section 3.5 présente un état de l'art sur les mécanismes proposés pour la vérification et l'analyse des politiques de sécurité IPsec. Quant à la gestion des clés et la mise en place des associations de sécurité, le protocole IKEv2 est présenté dans la section 3.6, où nous présentons le fonctionnement et les échanges effectués par ce protocole. Enfin, nous concluons le chapitre dans la section 3.7.

3.1. Services de sécurité offerts par IPsec

Le protocole de sécurité Internet (IPsec) est un ensemble de protocoles qui sécurise la communication et l'échange des données sur un réseau IP. IPsec opère au niveau de la couche réseau. En raison de sa flexibilité et la transparence de ses applications, IPsec est largement utilisé pour établir des réseaux privés virtuels (VPN) [37] et même des réseaux VPN dynamiques et multipoints [117].

IPsec est un protocole qui propose la sécurisation de toute communication IP en fournissant ces services de sécurité. Il peut être mis en œuvre pour protéger le flux de

paquets IP entre une paire d'hôtes, une paire de passerelles ou entre un hôte et une passerelle (voir figure 3.1).

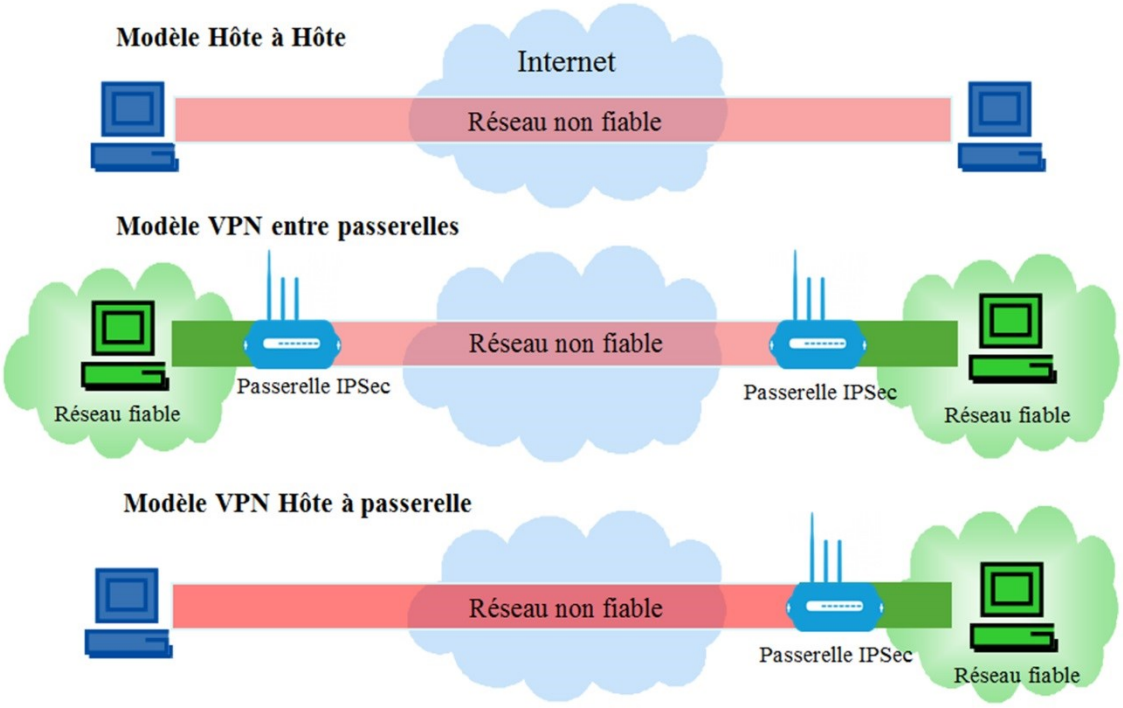

Figure 3. 1 Modèles de fonctionnement d'IPsec

IPsec assure les services de sécurité suivants.

- **Intégrité des données :** lors d'une communication sécurisée par IPsec, l'intégrité des données est vérifiée. IPsec vérifie que le contenu d'un datagramme IP n'a pas été modifié, délibérément ou en raison d'erreurs aléatoires.
- **Confidentialité des données :** la confidentialité transforme les données d'un format intelligible (texte en clair) en un format inintelligible (texte chiffré). Lors des communications sécurisées par IPsec, il garantit qu'un attaquant non autorisé ou une écoute indiscrète ne pourraient pas comprendre le contenu des paquets transmis.
- **Authentification de l'origine des données :** IPsec permet à un nœud quelconque de vérifier que chaque datagramme IP reçu a été émis par l'expéditeur revendiqué (source de données).
- **Authentification des entités :** ce service implique l'identification et l'authentification des entités communicante. L'identification fait référence à

une entité (utilisateur, équipement) revendiquant son identité en fournissant un identifiant (nom, pseudonyme, adresse IP, nom de domaine). L'authentification consiste à fournir l'identité revendiquée en fournissant un ou plusieurs éléments d'authentification, appelés informations d'identification.

- **Détection de rejeu :** consiste à détecter si les données reçues sont dupliquées à partir d'un échange précédent. Certaines données peuvent avoir été envoyées de manière sécurisée par une entité légitime ; mais ils peuvent être copiés et injectés à nouveau vers la même destination. Les données sont toujours authentiques, mais elles sont déjà traitées plusieurs fois.
- **Contrôle d'accès :** ce service permet d'empêcher l'utilisation non autorisée d'une ressource. Pour IPsec, le contrôle d'accès concerne un hôte ou une passerelle de sécurité. Seuls les hôtes autorisés ou les passerelles de sécurité destinée à établir des communications IPsec seront autorisés à le faire.

3.2. Modes de fonctionnement

Le protocole IPsec définit deux modes de fonctionnement différents, le mode transport et le mode tunnel. IPsec est censé sécurisé soit la totalité du paquet IP, ou seulement les charges utiles. Cette différence fait la distinction entre les deux modes. Lorsqu'un réseau entier est protégé avec IPsec, le mode tunnel est utilisé. Lorsque la communication se fait entre deux hôtes, le mode transport est le plus approprié. En outre, le mode d'imbrication (Nesting) est un mode hybride qui favorise l'utilisation des deux modes ensemble. Il s'agit bien d'encapsuler l'IPSec dans l'IPSec. La figure 3.2 présente un exemple d'encapsulation d'un paquet IPsec en mode transport et en mode tunnel.

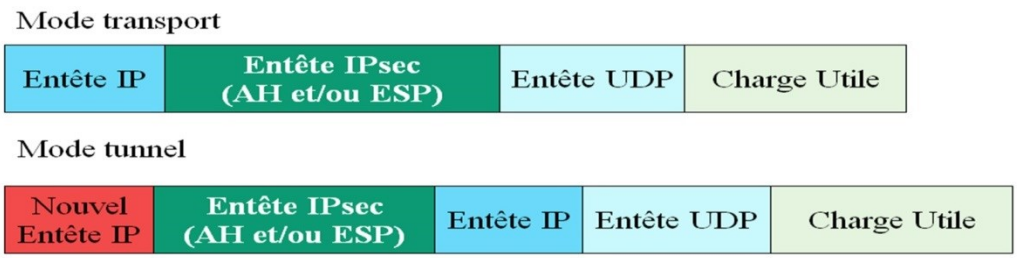

Figure 3. 2 Encapsulation IPsec en mode transport et mode tunnel

3.2.1. Mode transport

Ce mode est utilisé pour les communications de bout en bout tel que le modèle client/serveur. Il s'agit du mode par défaut d'IPsec. Le mode transport ne modifie pas l'en-tête initial; il s'intercale entre le protocole réseau (IP) et le protocole de la couche supérieure (transport : TCP, UDP...), par conséquent, seules les données provenant de cette couche sont cryptées. Donc, pour le mode de transport, l'en-tête IPsec est inséré entre l'en-tête IP et la charge utile IP du paquet source.

3.2.2. Mode tunnel

En mode tunnel, non seulement la charge utile est protégée, mais aussi l'en-tête IP. Ce mode ajoute un nouvel en-tête au début d'un paquet. Cet en-tête lui permet d'être transporté vers le tunnel d'extrémité, où l'en-tête d'origine va être restauré. L'en-tête IPsec encapsule le paquet IP source, et l'ensemble est encapsulé à son tour par un nouvel en-tête IP. Ainsi, le tunnel correspond à une structure de données dans laquelle un paquet IP contient un autre paquet IP. Le mode tunnel est utile pour la sécurisation des datagrammes qui traversent un réseau non sécurisé. IPsec nécessite l'utilisation du mode tunnel si au moins un autre point d'extrémité d'une association de sécurité est une passerelle qui protège les paquets qu'elle achemine.

3.3. Les sous-protocoles d'IPsec

IPsec utilise deux protocoles différents selon les besoins en sécurité des utilisateurs. Le premier est le protocole d'authentification (AH) [38], qui vise à établir l'identité des extrémités de façon certaine, sans pour autant garantir la confidentialité des données. Le deuxième protocole est le protocole de confidentialité (ESP) [39]. De plus de l'authenticité, ESP a pour but de chiffrer les données.

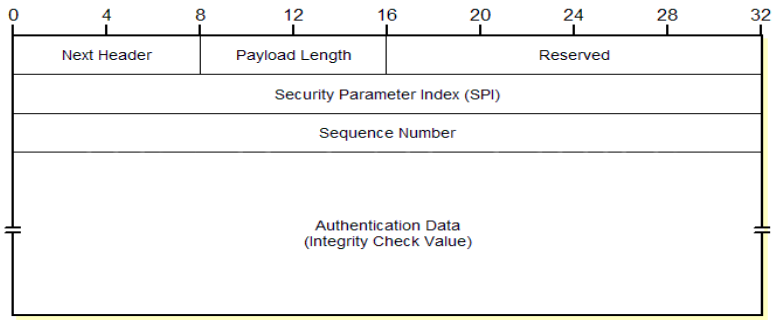

Figure 3. 3 Composition de l'entête AH [46]

3.3.1. Le protocole d'authentification AH

Le protocole AH fournit les services de sécurité suivants : l'intégrité en mode non-connecté, l'authentification des données et optionnellement la protection anti-rejeu. L'en-tête AH sert à transporter le résultat du contrôle d'intégrité sur tout le paquet d'origine. Comme aucun chiffrement n'est effectué, l'en-tête d'authentification n'assure pas la confidentialité. Par conséquent l'en-tête est beaucoup plus simple que celui d'ESP. Comme présenté dans la figure 3.3, l'en-tête AH se compose de six champs :

- Next Header « Suivant » : un champ qui identifie le type de l'entête suivant.
- Payload Length « Longueur » : définit la longueur de l'entête.
- Reserved « Réservé » : un champ réservé pour utilisations futures.
- Security Parametres Index (SPI) « Indice de paramètres de sécurité » : sert à identifie l'association de sécurité.
- Sequence Number « Numéro de séquence » : définit une valeur de compteur croissant.
- Authentication Data « Données d'authentification » : un champ variable de longueur multiple de 32 bits qui contient la valeur MAC.

Le protocole AH utilise des fonctions de hachage à clé pour calculer un condensé de message sur l'ensemble du datagramme IP (à l'exclusion des champs modifiables dans l'en-tête IP)[1]. Lorsque le nœud IPsec reçoit un paquet protégé par AH, il peut calculer le condensé et comparer le résultat à la valeur du paquet. Si ceux-ci correspondent, le récepteur peut être sûr que le paquet n'a pas été. Par conséquent, ce protocole permet la protection contre les attaques par rejeu, par usurpation d'identité et les attaques par déni de service, quand elles sont basées sur la charge impliquée par les calculs cryptographiques.

3.3.2. Le protocole de confidentialité ESP

En plus d'assurer l'intégrité, l'authentification des données et optionnellement la protection anti-rejeu, le protocole ESP assure également la confidentialité et la protection contre l'analyse de trafic. L'en-tête ESP est plus complexe que l'en-tête AH. ESP ajoute non seulement un nouveau entête entre l'entête IP et la charge utile

[1] Ces champs sont les suivants: Type de service, flags, décalage de fragment, durée de vie et somme de contrôle d'en-tête dans l'en-tête IPv4.

du paquet, mais il ajoute également certains champs à la fin du paquet. La figure 3.4 illustre le format d'un paquet ESP. Un paquet ESP définit les champs les suivants :

- Champs d'entête ESP :
  - Security Parameters Index (SPI) « Indice de paramètres de sécurité » : ce champ identifie l'association de sécurité utilisée.
  - Sequence Number « Le numéro de séquence » : une valeur de compteur croissant permet de détecter le rejeu de paquet IP.
- ESP Payload Data « charge utile ESP » : ce champ peut contenir soit un paquet IP complet (en-tête IP + extensions + données de niveau transport), soit l'extension destination suivie de données de niveau transport, soit des données de niveau transport uniquement [39].
- Les champs de la queue ESP :
  - Padding « bourrage » : permet d'aligner les données à chiffrer sur un nombre d'octets dépendant de l'algorithme de chiffrement utilisé.
  - Pad lenght « Longueur du bourrage » : indique la longueur en octets du champ bourrage.
  - Next Header « L'en-tête suivant » : identifie le type de données contenues dans le premier champ du champ charge utile.
- ESP Authentication Data « données d'authentification ESP » : ce champ est optionnel et sa présence dépend de l'association de sécurité utilisée.

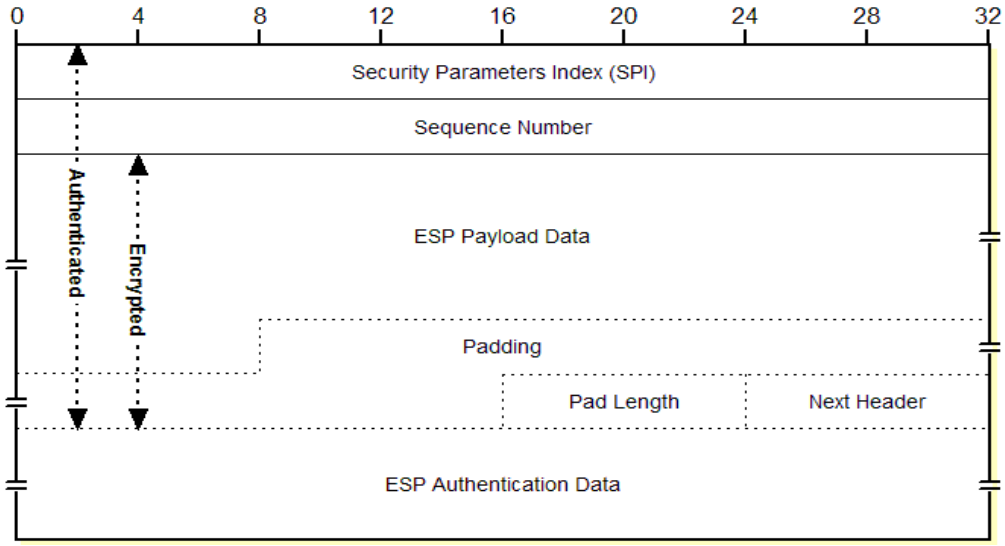

Figure 3. 4 Format de paquet ESP [46]

Donc, le fait que les champs ESP contiennent la charge utile d'origine, rend la sécurité plus complexe qu'avec AH qui n'ajoute qu'un seul en-tête. Le chiffrement sera appliqué aux données utiles jusqu'au champ NEXT. Les données d'authentification protègent les données du champ SPI au champ NEXT. ESP fournit une protection contre les attaques de type espionnage et autres divulgations d'informations et optionnellement contre les attaques par rejeu et par analyse du trafic. La figure 3.5 illustre l'encapsulation ESP en utilisant à la fois les modes de transport et le mode tunnel.

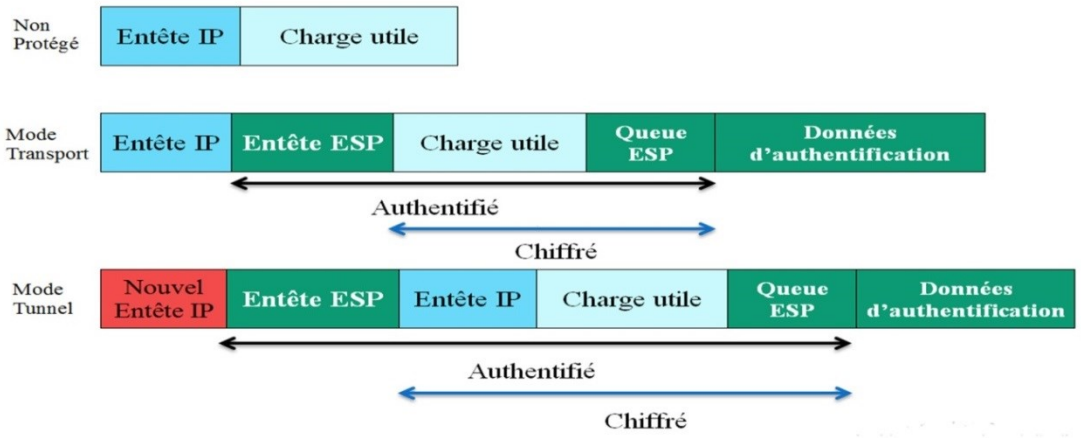

Figure 3. 5 ESP en mode transport Vs tunnel

Bien qu'ESP couvre les services offerts par AH, il faut souligner que le protocole AH offre des services plus complets que ceux du ESP. AH est le seul à protéger l'en-tête du paquet (en-tête original en mode Transport, en-tête IPSec en mode Tunnel), alors qu'ESP ne protège que les données (c'est-à-dire tout au plus l'en-tête original en mode Tunnel). Le tableau 3.1 présente une comparaison entre les différents services de sécurité offerte par chaque protocole.

	AH	ESP	ESP avec Authentification
Contrôle d'accès	✓	✓	✓
Intégrité des données	✓	×	✓
Authentification de l'origine des données	✓	×	✓
Détection de rejeu	✓	✓	✓
Confidentialité	×	✓	✓
Confidentialité du flot de trafic	×	✓	✓

Tableau 3. 1 Les services de sécurité offerts par AH Vs ESP

3.4. L'architecture de sécurité

L'architecture de sécurité IPsec définit deux types de bases de données : la première est la base de données des politiques de sécurité (SPD) et la deuxième est la base de données des associations de sécurité (SAD). La SPD contient un ensemble de politiques de sécurité sous forme de listes ordonnées. Ces listes contrôlent d'accès, elles sont configurés et mises à jour par un administrateur réseau. Elles contiennent des critères de contrôle d'accès, similaires à des règles de Pare-Feu. L'objectif d'une politique de sécurité est de définir le comportement approprié face aux paquets IP entrants et sortants. La politique spécifie le trafic qui doit être protégé avec IPsec, et celui qui doit être contourné ou passé sans protection. Tous les paramètres concernant la SPD pourraient être configurés manuellement par un administrateur ou configurés d'une façon automatique. Cependant, cela nécessite un réordonnancement à la volée des entrées et une gestion plus sophistiquée, afin d'éviter les conflits des politiques de sécurité.

La SAD contient les associations de sécurité qui définissent des paramètres de sécurité permettant l'instanciation d'une communication IPsec. Son objectif est d'offrir des services de sécurité pour le trafic entrant et sortant. La SAD est configurée manuellement ou bien gérée dynamiquement grâce au protocole d'échange de clé Internet (IKE). Lorsque IKE est utilisé, (ce qui est généralement le cas), une base de données d'autorisation de paires (PAD) est utilisée. La PAD contient des informations sur les nœuds des réseaux autorisés à communiquer avec les nœuds IPsec, ainsi que le protocole et la méthode d'authentification. Lorsqu'un paquet IPsec doit être protégé mais qu'aucune SA n'est trouvée, le nœud émetteur vérifie dans la PAD si l'autre nœud est autorisé à communiquer via IPsec. Si oui, une nouvelle SA peut être établie dynamiquement au moyen du protocole IKEv2. De plus amples détails concernant IKE sont expliqués dans la section 3.5.

3.4.1. Les politiques de sécurité

Une politique de sécurité réseau est un ensemble d'exigences qui contrôlent le comportement des entités dans ce réseau. Ce comportement est défini par un ensemble de contraintes, qui sont destinées à régir : l'accès aux données, leur utilisation et leur transmission. Les exigences d'une politique de sécurité sont définies comme un ensemble de règles de filtrage; qui sont triées par un ordre

particulier, ce qui garantit la bonne exécution des directives de la politique. Généralement, les politiques de sécurité sont utilisées pour assurer trois fonctionnalités principales: la confidentialité (secret des données), l'intégrité (originalité des données) et la disponibilité (accès aux données) [41].

Une politique de sécurité IPsec est une liste ordonnée d'entrées contenant des critères de contrôle d'accès. Cette liste est définie comme une liste de contrôle d'accès (ACL) à déclencheur unique. Ceci signifie que l'ordre des enregistrements qu'elle contient est très important. Plusieurs règles peuvent s'appliquer à un même paquet, mais seule la première sera réellement appliquée, d'où l'importance de l'ordre. Chaque implémentation IPsec contient deux SPDs, une pour le trafic entrant, l'autre pour le trafic sortant. Chaque entrée de ces SPDs précise un traitement (action) à appliquer au paquet correspondant. Ces traitements sont DENY « rejette », BYPASS « passer sans protection » ou bien PROTECT « protégé avec IPSec ».

- Rejeter un paquet : certains types de trafic sont interdits d'être envoyés ou reçus car ils sont considérés comme inhérents d'insécurité.
- Passer sans protection : certains types de trafic sont permis d'être envoyé et reçus en clair.
- Protégé par IPsec : un traitement IPsec est nécessaire pour ce type de trafic.

Afin d'identifier un type de trafic, une politique de sécurité IPsec définit les sélecteurs de trafic suivants :

- Adresses IP de source et de destination, masques, intervalles... etc.
- Ports de source et de destination.
- Protocole supérieur (TCP/UDP)
- Utilisateur ou identifiant de système

Par défaut, la SPD est statique. En raison de l'importance de l'ordre des enregistrements qu'elle contient, un (des) administrateur (s) configure (nt) les listes de politique de sécurité selon les besoins du réseau. Cependant, les erreurs de configuration et l'incohérence entre les règles de la politique produisent des erreurs ou ce qu'on va appeler dans le reste de cette thèse « les conflits ». Ces conflits sont le sujet de notre première contribution présentée dans le chapitre V. De plus amples

détails concernant la gestion des politiques de sécurité sont expliqués dans la section 3.5.

3.4.2. Les associations de sécurité

Une association de sécurité (SA) est une relation entre deux ou plusieurs entités, elle décrit comment sécuriser les communications et assurer les différents services de sécurité. Cette relation est représentée par un ensemble d'informations qui peut être considéré comme un contrat entre les entités [42].

Une association de sécurité IPsec est une structure de données qui contient toutes les informations requises pour caractériser et échanger des données à protéger. Comme une SA est unidirectionnelle, deux SAs sont nécessaires afin de protéger les deux voies d'une communication IPsec bidirectionnelle. L'association IPsec s'appuie sur les protocoles AH et ESP pour offrir les différents services de sécurité. Chaque SA identifie trois éléments essentiels :

- Un indice de paramètres de sécurité (SPI) : Le SPI permet à la destination de sélectionner la SA correcte, sous laquelle le paquet reçu sera traité. C'est un bloc de 32 bits, envoyé en clair avec le paquet par l'expéditeur. Un SPI de valeur 0 indique l'absence d'une association de sécurité. Jusqu'à l'expiration de la SA, le même numéro SPI est utilisé par les deux côtés de la communication IPSec.
- L'adresse de destination : identifie la direction de l'application de l'association de sécurité.
- Identifiant de protocole de sécurité (SPId) : identifie le protocole de sécurité utilisé par une association de sécurité (ESP ou bien AH).

En outre, chaque SA contient également d'autre champs qui précisent le type de paquets sur lequel elle s'applique afin de garantir la granularité des données[2] .

Une SA définit les paramètres suivants :

- L'adresse IP source.

[2] La granularité des données désigne leur niveau de détail. Plus la granularité est fine, plus la donnée est détaillée et son analyse précise.

- Les ports source et destination : qui permettent de limiter la SA à un certain type de trafic voire à une session.
- Un nom sous forme standard (X500 ou DNS) ce qui permet entre autres d'avoir des SAs dédiées à des utilisateurs/hôtes.
- L'algorithme utilisé pour l'authentification, ainsi que les éventuels clés publiques associés.
- L'algorithme utilisé pour le chiffrement, ainsi que les éventuels clés publiques associés.
- Données et paramètres d'anti-rejeu : nombres de séquence, compteurs divers, fenêtres d'anti-rejeu… etc.
- La durée de vie : présentée par une unité de temps ou sous forme d'octets protégés, qui identifient le temps ou la quantité maximale de données à protéger
- Type d'en-tête IPSec : indique le mode de fonctionnement (Transport ou Tunnel.).
- Options de fragmentation : ensemble de paramètres utilisé pour effectuer la fragmentation des messages.
- Le lien vers la SPD : identifiant qui indique la correspondance dans la SPD à partir de la SAD.

3.4.3. Gestion du trafic

L'architecture de sécurité IPsec gère le trafic IP entrant et sortant grâce à l'interaction entre la SPD et la SAD.

3.4.3.1. Traitement des paquets entrants

À la réception d'un paquet entrant, le nœud IPsec récepteur examine tout d'abord l'en-tête du paquet pour savoir si un ou plusieurs services IPsec sont appliqués. Ensuite, il détermine la SA appropriée dans la SAD sur la base des trois champs suivant : l'adresse IP de destination, le protocole de sécurité (ESP ou AH) et le SPI.

Après l'identification de la SA correspondante (Si aucune SA valide n'est identifiée, le paquet est rejeté.), il récupère les paramètres nécessaires pour le traitement de ce paquet. Il vérifie ensuite le numéro de séquence pour confirmer que le service anti-

rejeu est activé. En outre le paquet doit aussi passer le contrôle d'authentification. Une fois le paquet est vérifié ou déchiffré, il consulte la SPD pour savoir si l'association de sécurité appliquée à ce paquet correspond à celle requise par la politique de sécurité, dans le cas inverse le paquet est rejeté. La figure 3.6 illustre le traitement d'un paquet IPsec entrant.

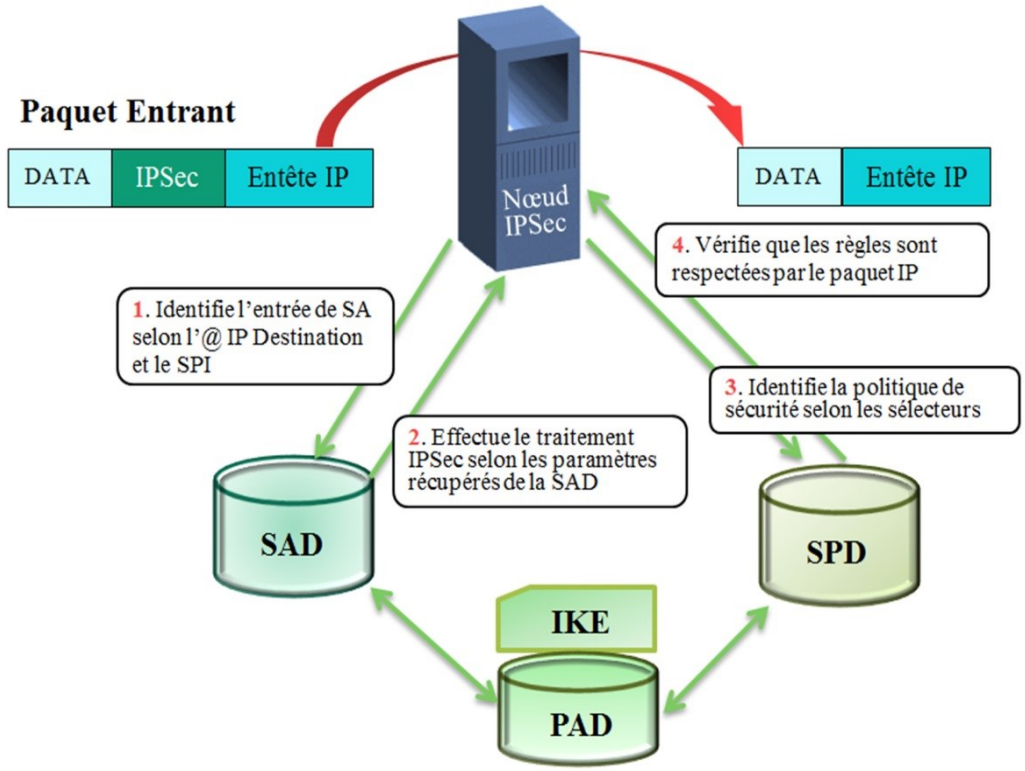

Figure 3. 6 Traitement IPsec d'un paquet entrant

3.4.3.2. Traitement des paquets sortant

Dans le cas d'un paquet sortant, le nœud IPsec émetteur, consulte tout d'abord la SPD pour identifier la politique de sécurité associée à ce paquet. Si la SPD indique qu'il est nécessaire d'appliquer une protection IPsec sur ce type de paquet, il va consulter la SAD pour identifier l'association de sécurité correspondante. Sinon, le paquet est soit envoyé sans protection ou bien rejeté.

Dans le cas où aucune SA n'est pas trouvée, il va initier une nouvelle association en faisant appel au protocole IKE. Ce protocole va négocier une nouvelle SA avec le nœud destinateur, après avoir vérifié dans la PAD que ce dernier est une paire autorisée. Après la création de la SA, le nœud émetteur applique le traitement IPsec

convenu avec le nœud récepteur et envoie le paquet. La figure 3.7 illustre le traitement IPsec d'un paquet sortant.

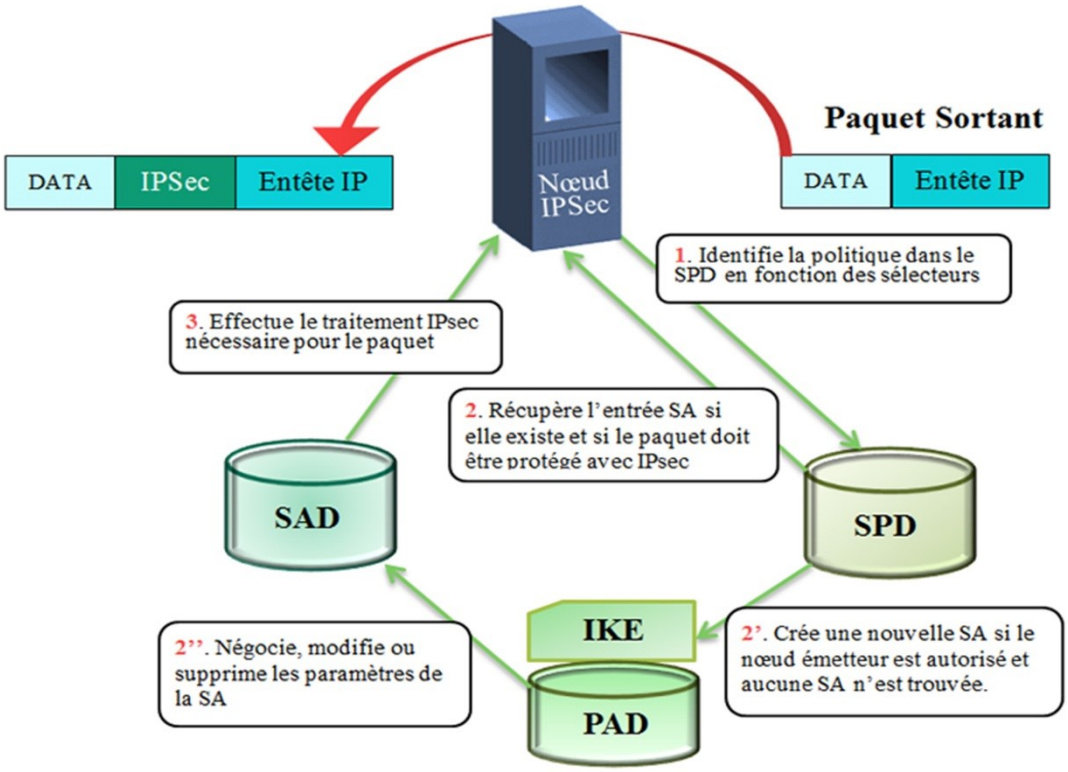

Figure 3. 7 Traitement IPsec d'un paquet sortant

3.5. Mécanismes de vérification des politiques de sécurité

Les politiques de sécurité peuvent être statiques, donc configurées et vérifiées manuellement par les administrateurs réseau. Cependant, IPsec est actuellement utilisé pour les réseaux dynamiques à grande échelle et aux environnements de systèmes embarqués tels que l'IoT. En raison des conditions dynamiques imposées par ces réseaux, telles que les mises à jour fréquentes des politiques, la configuration manuelle est sans doute inefficace [40].

Dans de tels réseaux, la vérification de la politique de sécurité devient plus difficile. Les scénarios Handover[3] présentent l'une de ces difficultés. En outre, la non-correspondance des adresses IP provoquée par le changement des adresses IP des utilisateurs finaux qui émergent d'un sous-réseau à un autre est l'une des principales

[3]Il s'agit d'un processus permettant d'obtenir un service continu lorsque l'utilisateur se déplace entre les cellules. Pendant le transfert, il y a une commutation entre les réseaux. Ce mécanisme transfère les utilisateurs vers un autre réseau ou une autre station de base.

causes de conflits. Par conséquent, La difficulté de vérification des politiques de sécurité s'aggrave lors de l'utilisation d'IPsec dans des environnements dynamiques. L'efficacité du protocole peut être compromise en raison des conflits produits dans ces politiques de sécurité. Le problème des conflits de politique de sécurité a fait l'objet de plusieurs études dans différents domaines, tels que : le cloud computing [43] et le domaine des réseaux sociaux [44]. Cependant, ce problème n'a pas été étudié de manière approfondie dans la littérature d'IPsec. La plupart des études IPsec se sont concentrées sur la mise en œuvre et l'optimisation du protocole, tandis que la politique de sécurité joue un rôle crucial pour le bon fonctionnement du protocole. Les travaux existant ont été déployés pour analyser les politiques et pour développer des techniques de détection des conflits. Cependant, ils manquaient de techniques efficaces pour la résolution des conflits. La plupart de ces travaux ont supposé un environnement statique idéal, ce qui n'est pas le cas d'une politique non-statique.

Dans cette section, nous examinons les recherches existantes dans ce domaine. L'étude de ces recherches a fait le sujet d'un papier scientifique qui a été acceptée et présentée à la conférence internationale «International Conference on Signal, Image, Vision and their Applications (SIVA 2018) », ensuite publiée dans la bibliothèque digitale du IEEE [C2].

Dans ce qui suit, nous présentons ces recherches et nous mettons en évidence les approches les plus importantes tout en soulignant les différents avantages et inconvénients de chaque approche.

3.5.1. Gestion des politiques de sécurité IPSec

Après la définition des directives et des exigences de sécurité du réseau, vient la spécification des règles de filtrage de trafic. Cette phase est appelée configuration de la politique de sécurité, généralement compliquée et sujet aux erreurs. Les conflits peuvent entraîner des failles de sécurité et des attaques à haut risque. Ces conflits peuvent être le résultat d'une mauvaise configuration ou d'une incohérence entre les différentes règles de la même politique (Conflits Intra-politique) ou entre des politiques différentes. (Conflits Inter-Politique).

Afin de garantir l'application correcte des directives prédéfinies, les conflits doivent être évités ou au moins identifiés afin de les supprimer. Cette solution n'est pas aussi

simple qu'il y paraît, en raison de nombreuses difficultés telle que ; le nombre croissant d'applications Internet, la nature des réseaux distribués, les différents types de contrôles de sécurité et le grand nombre de règles qui peut provoquer un nombre extrêmement élevé de conflits. Ces difficultés font de la gestion des politiques de sécurité une tâche très difficile pour les administrateurs. Par conséquent, il en résulte la nécessité de trouver des mécanismes plus adaptées pour la vérification des politiques de sécurité.

3.5.2. Les mécanismes proposés pour la vérification des politique IPsec

Plusieurs études ont été menées pour surmonter les problèmes des conflits et des erreurs de configuration. La vérification des politiques consiste à surveiller les changements de comportement ou les violations de sécurités provoquées par les conflits. Dans notre travail [118], nous avons mené une enquête sur les approches connexes qui ont été proposées pour la gestion des politiques de sécurité ainsi que les techniques de détection et de résolution des conflits.

Le concept d'analyse de politique de sécurité a été adopté pour IPsec initialement par Fu et al. [45]. Les auteurs définissent un conflit dans le cas où les implémentations de la politique ne satisfont pas les exigences souhaitées. Outre la détection des conflits, les auteurs proposent un mécanisme de récupération qui vise à remplacer la mise en œuvre des politiques. Cependant, leur proposition présente certains inconvénients, comme l'utilisation d'exigences de sécurité de haut niveau. De plus, la modification de ces exigences entraîne la régénération exécutive de la politique à chaque fois qu'une anomalie est détectée.

3.5.2.1. Approches pour la classification des conflits

La première classification des conflits IPsec a été présentée dans [46]. Les auteurs définissent les conflits d'une manière assez simple ce qui rend la mise en œuvre beaucoup plus facile. Une politique IPsec est considérée comme une liste de contrôle d'accès (ACL) et une liste de chiffrement (EL). Les auteurs classent les anomalies pour ACL et EL dans des scénarios intra et inter politique. Les auteurs ont conclu que ces anomalies pourraient être à l'origine de failles de sécurité. D'où la difficulté de prévention et de détection des intrusions. Cette classification a été ensuite améliorée et détaillée dans [47].

Une autre classification plus détaillée a été représentée dans [48], ou les auteurs ont présenté une taxonomie complète des conflits des politiques de sécurité IPsec. Les conflits IPsec ont été divisés en quatre types de conflits intra-politiques qui sont : l'observation, la corrélation, la généralisation et la redondance.

3.5.2.2. Approches pour la détection et la résolution des conflits

Le schéma proposé par Fu et al. a été formalisé en [49]. Les auteurs proposent une méthode de détection des conflits. Deux types de conflits sont définis, les conflits de sessions, qui se produisent lorsque plusieurs sessions IPsec sont établies pour remettre un paquet à plusieurs hôtes. Sachant que ce paquet est remis à l'hôte le plus éloigné avant le plus proche. Le deuxième type de conflit est le conflit multi-transformation, qui résulte de l'application d'une protection plus faible à un trafic déjà encapsulé. Les auteurs utilisent également le diagramme de décision binaire (BDD) pour comparer les règles traduites en fonctions booléennes. Le principal inconvénient de cette méthode est qu'elle se limite à la détection des conflits uniquement sans aucun processus de récupération. De plus, le traitement des règles de politique consomme beaucoup de temps, ce qui est inefficace pour un environnement dynamique.

Sun et al. [50] ont proposé une architecture qui stocke toutes les politiques IPsec dans un centre, accessible par un gestionnaire et géré par une politique de contrôle d'accès. Ils définissent une implémentation IPsec comme un programme qui peut établir des canaux IPsec et l'accès aux bases de données; comme Strongswan et Openswan. Le principal ajout de ce document est l'utilisation du centre, afin de prévenir les conflits et d'éviter l'accès aux bases de données par des implémentations non-autorisées. Cependant, ce mécanisme de résolution n'est fait que pour les conflits de diffusion, qui est la définition des conflits Inter-politiques par les auteurs.

L'aspect de performance et d'optimisation de temps d'exécution a été pris en compte pour la première fois par [51]. Les auteurs proposent un algorithme pour la détection et la résolution des conflits des politiques IPsec. L'algorithme proposé utilise un diagramme de décision binaire (BDD) pour représenter les politiques IPsec et découvrir la présence de conflits. L'idée de détecter les conflits dans cet algorithme est basée sur l'évaluation d'un certain nombre d'expressions booléennes, chacune d'elles représentant un type spécial de conflit de politique. Les variables de ces

expressions booléennes proviennent de l'ensemble S des fonctions de sommation et également de la nouvelle expression booléenne de règle qui va être traitée. Les fonctions de sommation sont calculées à partir de la liste complète des conditions de règle qui adoptent une approche dynamique pour gagner en performances et réduire la complexité. L'algorithme principal de cette approche est présenté à la figure 3.8.

```
//Analyse Policy P1 and Generates conflict-free Policy P2
Algorithm IPCDR(P1,P2)
{
    while(P1.GetRule(R))
    {
        if (P2 is empty)
        {
            //Place R in position 1 of list
            P2.AddRule2List(R,1);
            P2.S[R.Action] = R.Cond;
            return 1;
        }
        Check_non_conflicting(P2,R);
        Check_Approving_Redundancy(P2,R);
        Check_Conflicting_Redundancy(P2,R);
        Check_PartialOverlap(P2,R);
        Check_ProtectAction(P2,R);
    }
    return P2;
}
```

Figure 3. 8 Algorithme de vérification des politique SA proposé par Niksefat et al. [51]

L'algorithme prend une ancienne politique et génère une politique sans conflit de manière dynamique : les règles sont extraites une par une de l'ancienne politique et insérées dans la nouvelle politique. L'algorithme invoque des fonctions de détection et de récupération des conflits. Cependant, leur approche se limite aux conflits intra-politiques seulement.

3.5.2.3. Autres approches

Puisque les approches de vérification des politiques de sécurité IPsec ne sont pas si courantes, les approches proposées pour d'autres systèmes de contrôle de sécurité, tels que le pare-feu, constituent un arrière-plan intéressant pour ce travail, car ils ont une nature similaire de politiques. Par conséquent, nous donnons un bref aperçu de quelques études utiles dans ce domaine.

Le concept de vérification des politiques de sécurité des pare-feu a d'abord été introduit par AL-shaer et al. [52]. Ils ont proposé une classification des relations existantes entre les règles dans la même politique de pare-feu. La navigation itérative des politiques de sécurité fait partie des inconvénients de leur approche. Il s'agit d'une méthode très longue et inefficace pour les environnements dynamiques.

Un outil pour l'analyse des politiques de pare-feu appelé «FIREMAN» a été proposé dans [53]. Ce travail fournit une analyse des paquets intra-politique et vérifie la mise en œuvre correcte sur la politique de bout en bout. La principale limitation de cette approche est qu'elle détecte l'anomalie sans identifier les règles impliquées. Des travaux de suivi ont prouvé que la découverte de conflits n'est pas suffisante.

Différents outils de résolution des conflits ont été proposés. Une proposition d'un système pour la détection des conflits des pare-feu à l'intérieur des réseaux dynamiques, a été présentée dans [54]. Ce travail présente un modèle de détection et de résolution de conflits en temps réel.

Des recherches ultérieures ont développé de nouvelles techniques de résolution telles que la technique de priorisation [55]. Cette technique classe les conflits en fonction de leur niveau de gravité. Un autre exemple a été donné dans [56]. Les auteurs ont proposé une technique de résolution basée sur les techniques de première règle de correspondance (FMR) et dernière règle de correspondance (LMR).

Toutes les méthodes de résolution des conflits mentionnées ci-dessus dépendent de l'administrateur réseau, ce qui peut être difficile et sujet aux erreurs. Les auteurs de [57] se sont concentrés sur ce défi et proposent une solution alternative pour compenser l'intervention humaine. Ils utilisent un moteur de requête pour l'analyse des politiques de sécurité. Leur proposition vise à automatiser l'ensemble du processus de résolution des anomalies, sans se référer à l'intervention de l'administrateur. Néanmoins, leur méthode n'est pas évolutive et nécessite une mise en œuvre de politique de haut niveau.

Lorsque IPsec est utilisé dans des réseaux d'entreprises IPsec ou dans un environnement IoT, des centaines d'hôtes et d'appareils sont distribués sur le réseau [58]. Par conséquent, la topologie du réseau doit être prise en considération ainsi que l’aspect des performances. Le temps d’exécution, l’occupation d’espace mémoire et

l'efficacité sont des critères importants qu'on doit prendre en compte lors de la configuration des politiques. Ceci nous a motivés a proposé un nouvel algorithme pour la gestion automatique des politiques de sécurité IPsec. Notre proposition garantit à la fois la détection et la résolution des conflits. Plus de détail sur notre contribution sont présenté dans le chapitre V.

3.6. Protocole d'échange des clés internet : IKE

Les services de sécurité offerts par IPsec dépendent essentiellement de son protocole d'échange de clé sous-jacent. IPsec utilise le protocole IKE (Internet Key Exchange) pour établir et gérer les SAs et pour distribuer les clés partagées. Le protocole IKE est une partie essentielle dans la mise en œuvre d'IPsec. Ce protocole est utilisé pour négocier les clés secrètes et les paramètres de sécurités nécessaires pour les communications. La négociation se fait entre deux entités notées « Emetteur » et « Récepteur ». Le résultat de cette négociation est une clé sécrète utilisée pour créer une SA.

La première version du protocole IKEv1 proposé en 1998 [59] est actuellement remplacée par la deuxième version IKEv2 [60]. Depuis sa sortie, plusieurs modifications ont été adaptées pour IKEv1 [61]. Une clé partagée est utilisée dans le protocole IKEv1 proposé par Cheng [62]. Une clé secrète doit être partagée entre les deux pairs communicants avant que la négociation IKE commence. Cependant, IKEv1 et ses documents associés ont été obsolètes par IKEv2 en décembre 2005. Tous les changements qui ont été proposés et qui sont en cours dans IKE ont apporté une contribution positive en termes de simplicité, de flexibilité et de sécurité [63].

- **Simplicité** : IKEv2 réduit le nombre de phases et le nombre de messages à échanger par rapport à IKEv1.
- **Flexibilité** : dans IKEv2, les hôtes finaux peuvent choisir un sous-ensemble de sélecteur du trafic proposé par l'initiateur. Alors que dans IKEv1, ces sélecteurs ont été fixés et il n'y avait pas de choix pour les hôtes finaux sur la sélection des paramètres. En plus, Dans IKEv1, les durées de vie des SAs des ont été négociées au début et les deux parties devaient respecter cet accord. Dans IKEv2, il y a eu une plus grande flexibilité et chaque partie est capable de choisir une durée de vie SA de son choix indépendamment de l'autre. Le

résultat de ce changement est que les parties ayant des besoins différents d'une communication sécurisée peuvent mieux s'intégrer qu'auparavant.

- **Sécurité** : la notion de cookie a été améliorée dans IKEv2. Un mécanisme proposé par Pearlman [64] a été ajouté dans IKEv2. Si une partie soupçonne que les demandes entrantes ne sont pas authentiques, elle peut vérifier l'existence d'adresses IP réelles et correctes à partir desquelles la demande est générée. Ainsi, dans le cas où un répondeur reçoit un grand nombre de demandes IKE_SA à demi-ouverture, il doit les rejeter, sauf s'il contient une charge utile de notification (cookie).

3.6.1. Fonctionnement du protocole IKEv2

IKEv2 effectue une authentification mutuelle, négocie les clés secrètes et établit une association de sécurité entre deux paires. Cette association notée « IKEv2_SA » comporte les informations secrètes partagées permettant de sécuriser le trafic échangé entre ces deux paires. Les associations de sécurité pour les protocoles AH ou ESP sont notées « CHILD_SA ». Toutes les communications d'IKE sont basées sur un échange de paires de messages. Chaque échange comporte une demande et une réponse. Dans le cas où une demande n'a pas reçu sa réponse, elle sera retransmise pour des raisons de fiabilité. Afin de bien comprendre le fonctionnement d'IKEv2, il est important de définir quelques éléments nécessaires utilisés par le protocole, comme :

- **Identifiant de message** : chaque message IKE contient un Identifiant de message (ID) de 32 bits protégé cryptographiquement. Cet identifiant fournit une protection contre les attaques par rejeu. Il est mis à zéro pour la première demande dans chaque direction. Sinon, l'ID s'incrémente au fur et à mesure des demandes générées et reçues.
- **Nonce** : le rôle principal des nonces est de rafraîchir la dérivation de clé. Dans IKEv2, les nonces sont choisis d'une manière aléatoire avec une taille de 128 bits minimum.
- **Sélecteur du trafic** : ces charges utiles spécifient les critères de sélection des paquets qui seront acheminés via la nouvelle association de sécurité configurée. IKEv2 permet au répondeur de choisir un sous-ensemble du trafic proposé par l'initiateur.

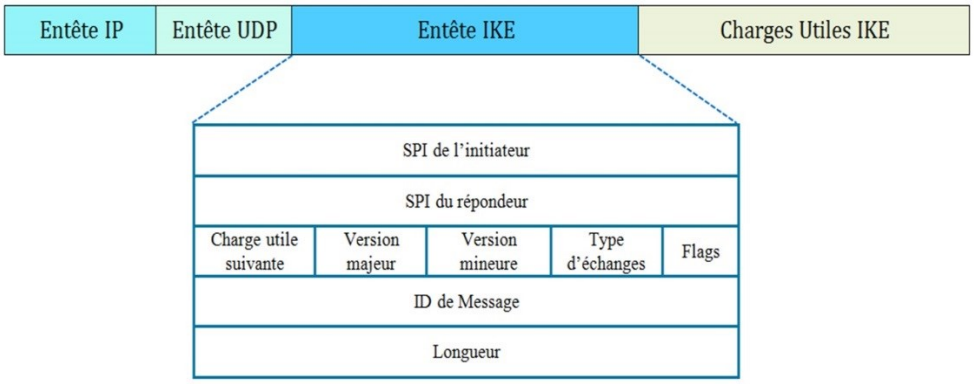

Figure 3. 9 Structure de l'entête IKEv2

3.6.1.1. L'entête IKEv2

Chaque message IKE commence par l'en-tête IKE, noté HDR. L'en-tête est suivi d'une ou plusieurs charges utiles IKE. Chacune de ces charges est identifiée par un champ nommé « Charge utile suivant » dans la charge qui la précède. La structure de l'en-tête IKE est présentée dans la figure 3.9. Le SPI du destinataire dans l'en-tête identifie une instance d'une association de sécurité IKE. Les différents champs de l'entête IKE sont définis dans le tableau 3.2.

Champ	Description
SPI Initiateur /Répondeur	l'index de paramètre de sécurité d'initiateur, une valeur de 8 octets qui sert à identifier une association de sécurité IKEv2 unique.
Charge utile suivante	un octet qui indique le type de la charge utile qui suit immédiatement l'entête en cour
Version Majeure/Mineure	4 bits qui indiquent la version majeure / mineure du protocole IKE utilisé
Type d'échange	un octet qui indique le type d'échange utilisé
Flags	un octet qui indique les options spécifie pour ce message
ID de message	4 octets qui indiquent l'identifiant d'un message et contrôlent la retransmission des paquets perdus en faisant la correspondance entre les demandes et les réponses
Longueur	4 octets qui indiquent la longueur du message (entête + charge utile).

Tableau 3. 2 Description des champs de l'entete IKEv2

3.6.1.2. Négociation des algorithmes cryptographiques

Dans le contexte d'IKE, il existe quatre types de paramètres cryptographiques qui doivent être négociés : l'algorithme de chiffrement (AES256, AES192, AES128, 3DES), l'algorithme de protection de l'intégrité (SHA384, SHA256, SHA1, MD5), le groupe Diffie-Hellman (DH) et la fonction pseudo aléatoire (PRF). La fonction PRF est utilisée pour générer des clés pour tous les algorithmes cryptographiques utilisés durant les échanges IKE. Le type de charge utile SA présente une hiérarchie qui indique le choix de protocoles IPsec (ESP et/ou AH).

La charge utile SA contient un ou plusieurs propositions, chaque proposition contient un ou plusieurs protocoles et chaque protocole contient un ou plusieurs transformations (ENCR_3DES, ENCR_AES, AUTH_HMAC_MD5, AUTH_HMAC_SHA). Chaque transformation contient plusieurs attributs qui spécifient un algorithme cryptographique. Par conséquent, le répondeur choisi un sous-ensemble parmi les propositions suggérées par l'initiateur. Toutefois, la réponse doit contenir les mêmes protocoles dans le même ordre que la proposition. En outre, le répondeur doit accepter une seule proposition ou toutes les rejeter et renvoyer un message d'erreur.

3.6.1.3. Génération des clés

Dans IKEv2, les clés sont dérivées à partir de l'algorithme PRF négocié, et puisque la quantité des clés nécessaire pour le chiffrement peut être supérieure à la taille du résultat obtenu par PRF, alors PRF peut être utilisé d'une manière itérative. Il existe plusieurs types d'algorithme de génération de clés :

- Les algorithmes avec des clés de taille fixe : acceptent n'importe quelle valeur choisie aléatoirement pour servir comme clé.
- Les algorithmes avec des clés de longueur variable : il faut préciser une taille fixe de clé comme étant une partie de la transformation cryptographique négociée.
- Les algorithmes pour lesquels toutes les valeurs ne sont pas des clés valides (DES, 3DES) : l'algorithme par lequel les clés sont dérivées à partir des valeurs arbitraires doit être spécifié par la transformation cryptographique.

- Les algorithmes de protection d'intégrité comme le HMAC (Hashed Message Authentification Code) : la taille de clé fixe est la taille du résultat de la fonction de hachage sous-jacente.

Les associations de sécurité IKE (IKE_SA), ESP et AH (Child_SA) utilisent des clés secrètes qui ne sont pas utilisées que pour une durée limitée et qui ne protège qu'une partie limitée de données. Cela limite la durée de vie de toute l'association de sécurité. La nouvelle SA doit être mise à jour avant l'expiration de la SA précédente. Ce processus de rétablissement d'associations de sécurité est appelé changement de clé ou Rekeying en anglais.

3.6.2. Les échanges d'IKEv2

Le protocole IKEv2 a deux objectifs essentiels : premièrement, établir une association de sécurité IKE (IKE_SA) entre deux entités communicantes (initiateur et répondeur). Deuxièmement, négocier et créer une ou plusieurs associations de sécurité enfants (CHILD_SA) qui fonctionnent avec les protocoles AH et ESP.

Les communications IKEv2 se réalisent en deux phases. La première phase commence toujours par un échange d'initialisation noté IKE_SA_INIT suivi par un échange d'authentification IKE_AUTH. La deuxième phase est utilisée lorsqu'on a besoin de plus de CHILD_SA, ainsi que pour échanger des messages informatifs. Le tableau 3.3 présente la description des différents paramètres utilisés par IKEv2.

Notation	Description
HDR	L'entête du message
SAi/ SAr	Définit la liste des algorithmes cryptographiques supportés /choisis par l'initiateur/ répondeur
KEi/ KEr	Les valeurs publique de Diffie-Hellman envoyés par l'initiateur/ répondeur
Ni/Nr	Les nonces, des valeurs aléatoire envoyés par l'initiateur/ répondeur
SK {…}	Fournit le chiffrement et l'intégrité des données
IDi/ IDr	Les informations d'identité d'initiateur/ répondeur
AUTH	Les informations d'authentification
CERT	Certificat
CERTREQ	Demande de certificat

Tableau 3. 3 Description des paramètres utilisés durant les échanges IKEv2

3.6.2.1. L'échange IKE-SA-INIT

IKE_SA_INIT correspond à la première phase de la communication. Il comporte une paire de messages utilisée pour négocier les paramètres confidentiels SAi et SAr entre l'initiateur et le récepteur. Il vise à échanger les valeurs publiques de Diffie-Hellman KEi et KEr, et les nonces Ni et Nr, ainsi pour négocier les algorithmes cryptographiques. Le résultat de cet échange est la génération des clés qui seront utilisées par la suite dans le deuxième échange. Les clés générées sont :

- Sk_ai/ Sk_ar : clés pour l'authentification (protection d'intégrité).
- SK_ei/ SK_er : clés pour cryptage.
- SK_pi/SK_pr : utilisés lors de la génération d'une charge utile AUTH
- SK_d : utilisé pour la dérivation de matériel de clé supplémentaire pour CHILD_SAs.

3.6.2.2. L'échange IKE-AUTH

La deuxième paire de messages IKE_AUTH authentifie les messages précédents, échange des identités et des certificats et établit la première CHILD_SA. Certains champs de ces deux messages seront cryptés et protégés par les clés établies dans le premier échange, de cette façon toutes les parties de ces messages sont authentifiés. Si l'authentification de l'identité est passée, les associations IKE_SA et CHILD_SA sont construite à la fin de cette phase. Les échanges de la première phase sont présentés dans la figure 3.10.

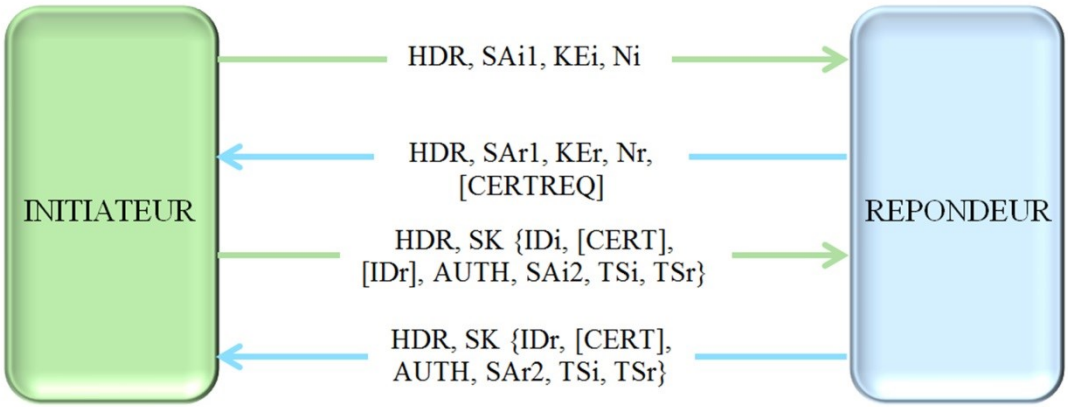

Figure 3. 10 Les échanges de la première phase : IKE_SA_INIT et IKE_AUTH

3.6.2.3. Création de CHILD-SA

L'échange de la deuxième phase peut être initié par l'initiateur ou bien le récepteur. De cette façon celui qui va commencer l'échange sera l'initiateur de la communication. Dans cette phase le rekeying est fait pour toutes les deux associations de sécurité IKE_SA et CHILD_SA. De plus, CREATE_CHILD_SA et d'autres échanges sont utilisés pour créer des nouveaux CHILD_SAs.

L'échange CREATE_CHILD_SA consiste en une seule paire requête/ réponse. Donc, Une CHILD_SA est créée en envoyant une demande CREATE_CHILD_SA. Le message qui suit l'en-tête est crypté et le message comprenant l'en-tête est protégé en intégrité à l'aide des algorithmes cryptographiques négociés pour IKE_SA. Après la réception de cette demande le répondeur utilise le même identifiant du message pour répondre à l'initiateur avec une réponse contenant les offres d'association de sécurité acceptées et les valeurs de Diffie-Hellman choisis (KEr). Cet échange est illustré dans la figure 3.11.

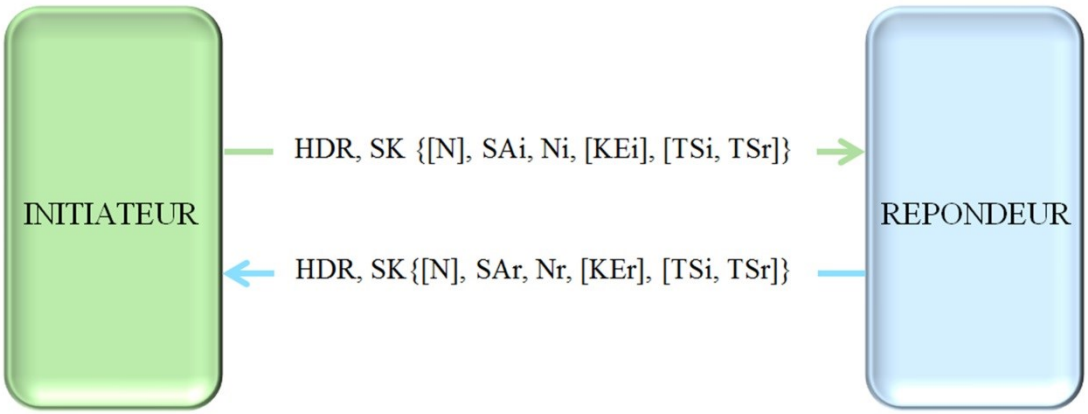

Figure 3. 11 Les échanges de la deuxième phase : CREATE_CHILD_SA

3.6.2.4. Les échanges Informatifs

Dans certains cas d'erreur ou d'événement particulier, les entités communicantes échangent des messages de contrôle les unes aux autres. Ce type de message est transmis avec l'échange informatif « INFORMATIONAL ». Tous les messages de ce type sont protégés avec les clés négociés durant l'échange initial. Dans un message informatif, la charge utile peut être vide comme dans le cas de test d'activité d'un point terminal. Cependant, plusieurs charges utiles peuvent être ajoutées aux

messages informatifs comme les charges utiles de notification, de suppression ou de configuration.

3.7. Conclusion

Dans ce chapitre, nous avons présenté le protocole de sécurité internet IPsec. Nous avons introduit les services de sécurité offerts par ce protocole. Ensuite, nous avons expliqué ces deux modes de fonctionnement ainsi que ces protocoles sous-jacents. En outre, nous avons examiné les éléments de base de son architecture de sécurité, comme les politiques de sécurité et les associations de sécurité. Un état de l'art sur les mécanismes proposés pour la gestion des politiques de sécurité IPsec a été présenté dans ce chapitre. Nous avons présenté les travaux qui ont été utile pour ce travail, et qui avaient motivé notre contribution. Finalement, nous avons présenté le protocole IKEv2 ; le protocole standard d'IPsec pour la création des associations de sécurité et l'échange des clés. Nous avons présenté un aperçu sur le fonctionnement du protocole et ses échanges. Nous avons constaté qu'IKEv2 est un protocole populaire et flexible, par contre il présente quelques failles de sécurité, comme sa vulnérabilité contre les attaques DoS, Replay et MITM. En outre, le protocole est assez complexe en termes de calcul et d'opérations cryptographiques. De ce fait, l'utilisation de ce protocole avec IPsec pour l'IoT présente quelques limitations. Ainsi, nous discuterons les solutions proposées pour la gestion des clés dans les systèmes IoT dans le chapitre suivant.

CHAPITRE IV

Authentification et échange de clé dans l'IoT

La communication machine à machine (M2M) joue un rôle essentiel dans l'Internet des objets (IoT). Elle permet aux systèmes sans fil et câblés de surveiller les environnements et d'échanger les informations entre les différentes machines sans intervention humaine. Afin d'exploiter les applications M2M et promouvoir le développement de l'IoT, l'Internet Engineering Task Force (IETF) a développé le standard 6LoWPAN pour permettre aux appareils M2M basés sur IP de se connecter à Internet. Bien que le protocole 6LoWPAN ait spécifié les issus importants dans les communications M2M, divers problèmes de sécurité n'ont pas été résolus. Les plus importants problèmes de sécurité qui menace les réseaux 6LoWPAN sont l'authentification mutuelle et le problème d'établissement et d'échange des clés.

Dans ce chapitre, nous présentons les solutions proposées dans la littérature pour l'authentification et la gestion des clés dans l'IoT. Tout d'abord nous introduisons les différentes notions que nous avons étudiées pour le besoin de cette thèse, y compris les concepts de base de la cryptographique : clés cryptographiques, techniques utilisées et les algorithmes recommandés pour l'IoT.

Ce chapitre est constitué de quatre sections. Section 4.1 définit les concepts cryptographiques de base. Nous commençons par la définition des clés cryptographiques et leur gestion. Ensuite, nous présentons les différentes techniques cryptographiques. La cryptographie symétrique basée sur la pré-distribution des clés secrètes, ainsi que la cryptographie asymétrique basée sur des clés publiques. En outre, nous soulignons les majeures déférences entre ces deux techniques. De plus, nous présentons les différents algorithmes utilisés par chaque technique, en donnant des exemples des algorithmes les plus utilisés dans le domaine d'IoT. La section 4.2 présente un état de l'art sur les différentes approches proposées pour assurer l'authentification des appareils IoT. Nous commençons par la définition de cette propriété de sécurité essentielle. Ensuite nous présentons les solutions proposées dans ce contexte. Enfin, nous présentons une comparaison entre les approches citées. La section 4.3 introduit un état de l'art sur les solutions de gestion et d'échange des clés dans l'IoT. Nous examinons les différents protocoles proposés dans la littérature. Ainsi, nous

divisons les solutions en deux catégories : solutions non-standards et solutions basées sur des protocoles standards. Ceci nous permettra de positionner notre travail par rapport à ces diverses approches dont nous nous sommes inspirés pour mener à bien notre travail. Dans les solutions basées sur des protocoles standards, nous concentrons sur les solutions inspirées du protocole IKEv2. Nous examinons ces protocoles afin de pouvoir les comparer avec notre contribution présentée dans le chapitre VI. Finalement, nous conclurons le chapitre dans la section 4.4.

4.1. Concepts de base de la cryptographie

De nos jours, la cryptographie est omniprésente, des mots de passe hachés au courrier chiffré, des réseaux privés virtuels IPSec et même des systèmes de fichiers chiffrés. Le but de la cryptographie ne se limite pas qu'à la dissimulation d'informations ; la cryptographie est utilisée pour assurer l'authentification, l'intégrité et la confidentialité. La mise en œuvre de la cryptographie implique la mise en œuvre d'une infrastructure à clé publique (PKI), notamment par l'introduction, en 1976, de la cryptographie à clé publique par Diffie et Hellman [65]. Une PKI définit les processus et les technologies utilisées pour implémenter un système cryptographique.

Dans cette section, nous commençons par discuter les concepts nécessaires pour comprendre la cryptographie. Nous présentons des rappels de mathématiques pour la cryptographie, puis nous examinons les outils cryptographiques de base, ainsi que les outils cryptographiques avancées.

4.1.1. Clés cryptographiques

Le paradigme révolutionnaire de l'IoT permet l'interaction entre les objets intelligents, diffusés de manière omniprésente sur Internet. La communication sécurisée trouve principalement ses racines dans la mise en œuvre des protocoles de gestion des clés (KMP) robustes. La gestion des clés implique la création, le renouvellement et l'échange de clés privées. C'est un élément essentiel de la sécurité des appareils. Des clés perdues ou compromises peuvent compromettre la sécurité d'un réseau entier. Les mécanismes de gestion des clés permettent à deux appareils distants de négocier certaines

informations d'identifications telles que les clés secrètes ou les clés de cryptage, qui sont utilisées pour protéger le flux d'informations entre ces appareils.

La clé est le support de la sécurité cryptographique. Elle doit rester secrète pendant toute la durée de vie du réseau, du déploiement à la révocation. La gestion des clés est très importante, elle implique le concept d'autorisation, car les informations d'identification et les clés de sécurité ne sont données qu'aux appareils capables de prouver leur légitimité. L'utilisation de ces clés définit le mode de cryptage d'un système cryptographique, s'il est symétrique ou bien asymétrique.

4.1.2. Techniques cryptographiques

Il existe deux catégories principales de cryptographie : la cryptographie symétrique, également appelée cryptographie à clé privée, et la cryptographie asymétrique, ou ce qu'on appelle la cryptographie à clé publique. En outre, les fonctions de hachage présentes une autre catégorie de cryptographie, qui pourrait être appelée la cryptographie sans clé [66]. En comparant la cryptographie à clé symétrique et la cryptographie à clé asymétrique, nous constatons qu'aucune d'entre elles n'est nécessairement meilleure, dans l'ensemble de toutes les situations. Chacune d'elles présente des points de force et de faiblesse selon la situation. Dans de nombreux cas, la cryptographie à clé symétrique est beaucoup plus rapide que l'asymétrique, Cependant, la cryptographie symétrique pose un problème d'échange de clés.

4.1.2.1. La cryptographie symétrique

La cryptographie symétrique basée sur la pré-distribution des clés représente l'approche la plus simple qui puisse être adoptée pour activer les services de sécurité dans l'IoT [67]. La force des algorithmes symétriques dépend de la sécurité d'échange de clé entre l'expéditeur et le récepteur. Ces algorithmes sont classés en chiffrements de blocs et chiffrements de flux. Le chiffrement de bloc est effectué bit par bit, alors que le chiffrement de flux est effectué par un groupe de 64 bits.

Le processus essentiel des algorithmes à clé symétrique est la génération de la clé. Ce processus implique des opérations mathématiques complexes. Un autre processus est le chiffrement, ce dernier consiste en des tours de chiffrement, chaque tour est basé sur

certaines fonctions mathématiques. Les algorithmes cryptographiques sont généralement conçus pour prendre en moyenne 10 à 20 tours. Ceci vise à maintenir le processus de chiffrement suffisamment fort pour répondre aux exigences du système. L'augmentation du nombre de tours assure une meilleure sécurité, mais augmente la consommation d'énergie [68].

On suppose qu'Alice envoie un message secret à Bob. Si Alice et Bob utilisent un système symétrique, Alice partage la clé utilisée pour le chiffrement avec Bob. Ce dernier utilise cette clé par la suite pour déchiffrer le message d'Alice. Cependant, si un adversaire malveillant intercepte la clé, il peut donc déchiffrer le message et accéder aux données confidentielles échangées durant la communication entre Alice et Bob.

En littérature, plusieurs approches symétriques ont été proposées pour sécuriser les communications dans l'IoT [69]. Dans ces communications, le nœud lui-même sert de nœud Internet, par conséquent, les calculs impliqués dans le processus de génération de clés doivent être également réduits dans la mesure où ils garantissent la sécurité nécessaire. Ainsi, les approches légères sont plus pertinentes dans le contexte d'IoT. Usman et al. ont proposé un algorithme de chiffrement léger nommé Secure IoT (SIT) [70], qui est un chiffrement par bloc de 64 bits. Rajesh et al. ont également proposer un nouvel algorithme de cryptage symétrique minuscule (NTSA). Cet algorithme offre une sécurité renforcée pour le transfert de fichiers texte via le réseau IoT. Il introduit des confusions de clés supplémentaires pour chaque cycle de cryptage. Malgré les avantages de la cryptographie symétrique, elle présente aussi des inconvénients. En plus des problèmes de la charge de calcul et la consommation d'énergie, la dégradation d'un seul appareil peut compromettre la sécurité de l'ensemble de réseau. En outre, la configuration d'une clé dédiée pour chaque couple d'appareils ne s'adapte pas à la taille du réseau [71].

4.1.2.2. La cryptographie Asymétrique

Au-delà des approches basées sur les clés symétriques, les mécanismes de chiffrement asymétriques sont de plus en plus pertinents. La cryptographie asymétrique utilise des clés publiques et des clés privées qui sont mathématiquement liées les unes aux autres.

La clé publique utilisée pour le cryptage est diffusée publiquement pour que quiconque puisse utiliser et crypter des données. Seul le destinataire a une clé privée utilisée pour déchiffrer le message. La cryptographie asymétrique assure la confidentialité des données, authentifie les participants et assure la non-répudiation. Ce système repose sur des algorithmes cryptographiques basés sur des problèmes mathématiques, tels que certaines factorisations entières, des logarithmes discrets et des courbes elliptiques. Les paires de clés, publique et privée, peuvent être générées facilement et peuvent être utilisées pour le chiffrement et le déchiffrement. La force de la sécurité réside dans le fait qu'il est extrêmement difficile de déterminer une clé privée, correctement générée, à partir de sa clé publique. De plus, les clés privées d'une longueur suffisante ne peuvent pas être brisées par des attaques de force brute[1].

Revenant à l'exemple d'Alice et Bob, cette fois, ils utilisent un système asymétrique. Alice chiffre le message avec la clé publique de Bob afin que ce dernier puisse le déchiffrer avec sa propre clé privée. Ceci offre un meilleur niveau de sécurité, car même dans le cas où un adversaire malveillant intercepte la clé publique de Bob, il ne pourra pas déchiffrer le message sans la clé privée.

Les approches asymétriques sont généralement basées sur des certificats [72] ou bien utilisent les identités [73]. Ainsi, la plupart de ces propositions sont basées sur l'algorithme Diffie-Hellman (DH) et les approches d'échange de clés Diffie-Hellman basé sur les courbes elliptique (ECDH). Sachant que l'algorithme DH fonde sa potentialité sur la difficulté de résoudre le problème du logarithme discret. L'ECDH négocie le secret via des primitives ECC et donc les mêmes niveaux de sécurité offerts par DH peuvent être atteints avec des clés plus courtes.

4.1.2.3. Les fonctions de hachages

Les fonctions de hachage représentent un troisième type de cryptographie à côté de la cryptographie symétrique et asymétrique. Les fonctions de hachage n'utilisent pas de clé,

[1] Une attaque par force brute est une méthode utilisée en cryptanalyse pour décoder des données sensibles. Les applications les plus courantes pour les attaques par force brute sont le craquage de mots de passe et le craquage de clés de chiffrement. Un attaquant essaie simplement d'utiliser différentes combinaisons de caractères jusqu'à ce que la bonne combinaison soit trouvée

mais créent à la place une valeur de hachage unique de longueur fixe, communément appelée condensé, basée sur le message d'origine. Par conséquent, toute modification du message modifie également la valeur du hachage. Les fonctions de hachage sont utilisées dans les applications qui fournissent l'authentification et l'intégrité. Ces fonctions sont des algorithmes qui convertissent des messages de taille arbitraire, en appliquant un ensemble de transformations pour qu'ils renvoient en sortie le condensé qui a une taille fixe.

Il existe de nombreuses fonctions de hachage tels que SHA-1, 2 et 3, MD5, RIPEMD, etc. La force de la fonction de hachage est mesurée par trois propriétés, à savoir la résistance à la collision, la résistance à la pré-image et la résistance à la deuxième pré-image [74]. Il existe différentes relations entre ces trois propriétés de résistance. Une fonction qui est résistante à la collision est nécessairement résistante à la deuxième pré-image, mais l'inverse n'est pas vrai. Cependant, une fonction peut être résistante à la collision mais non résistante à la pré-image et vice-versa. En outre, une fonction peut être résistante à la collision mais non résistante à la deuxième pré-image et vice-versa.

4.1.3. Algorithmes cryptographiques

Les algorithmes cryptographiques génèrent des clés sous la forme d'une série de bits, utilisés pour crypter et décrypter des informations. Les algorithmes de chiffrement symétrique (DES, 3DES, AES…) utilisent une seule clé pour le cryptage et le décryptage des messages. Cependant, un algorithme de chiffrement asymétrique (Diffie-Hellman, RSA…) utilise une clé publique pour crypter les données et une autre clé privée pour les déchiffrer. Ainsi, la clé publique peut être librement partagée. En revanche, la clé privée utilisée pour le déchiffrement doit être gardée secrète. Les fonctions de hachage n'utilisent pas du tout de clé, mais elles sont utilisées pour créer une empreinte digitale du message. Cette empreinte, théoriquement unique, détermine si le message a été modifié par rapport à sa forme d'origine.

4.1.3.1. Les algorithmes symétrique

Les algorithmes utilisés dans le cryptage à clé symétrique existent, pour la plupart, depuis de nombreuses années et sont bien connus. Tous les algorithmes utilisés

aujourd'hui sont complètement ouverts au public [75]. Ce type d'algorithme présente deux problèmes principaux. Le premier est de savoir comment s'assurer que l'expéditeur et le destinataire ont chacun la même clé. Ceci nécessite l'utilisation d'un service de messagerie ou d'un autre moyen de transport fiable. Le second présente le cas où le destinataire n'a pas la même clé pour déchiffrer le texte chiffré. Il existe des dizaines d'algorithmes symétriques actifs dans ce domaine. Dans ce qui suit, nous examinons les algorithmes les plus courants qui sont : Le Data Encryption Standard (DES), Le Triple Data Encryption Standard (3DES) et Advanced Encryption Standard (AES).

A. Le Data Encryption Standard (DES)

Le DES est devenu fonctionnel la première fois en 1976 aux États-Unis. Depuis, il a été utilisé par diverses parties dans le monde. Essentiellement, DES utilise une seule clé de 64 bits (56 bits de données et 8 bits de parité) et fonctionne sur des données en blocs de 64 bits. Cette clé est divisée en 16 sous-clés distinctes de 48 bits, une pour chaque tour, appelé « cycles Feistel » [76]. Chaque tour consiste en une phase de substitution, dans laquelle les données sont remplacées par des morceaux de la clé, et une phase de permutation, où les données substituées sont brouillées (réorganisées). Les opérations de substitution, parfois appelées opérations de confusion, se déroulent dans des S-box. De même, les opérations de permutation, parfois appelées opérations de diffusion, se déroulent dans des boîtes P. Ces deux opérations se produisent dans le "module F" du diagramme. La sécurité de DES réside principalement dans le fait que les opérations de substitution sont non linéaires. Donc, le texte chiffré résultant ne ressemble en rien au message d'origine. La figure 4.1 donne un schéma du fonctionnement de l'algorithme de chiffrement DES.

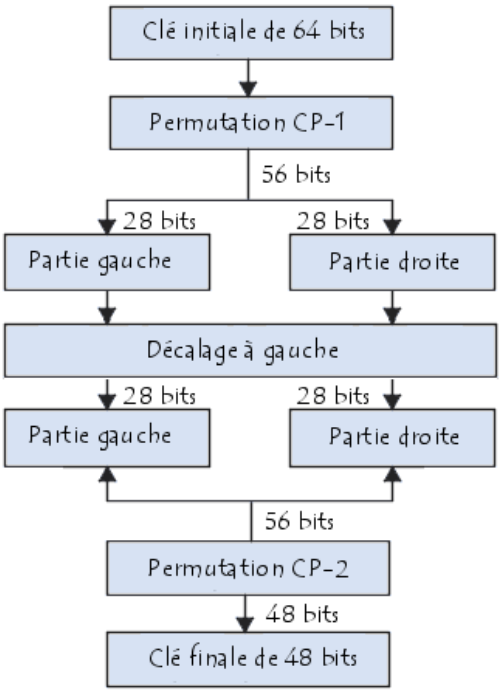

Figure 4. 1 fonctionnement de l'algorithme DES [76]

Bien que le DES ait été considéré très sûr pendant une certaine période de temps, il n'est plus considéré comme tel. Un projet d'informatique distribuée a été lancé en 1999 pour briser une clé DES. Le projet a réussi à le faire en un peu plus de 22 heures. Pour compenser cette faiblesse, 3DES a été créé. A la différence de DES, le 3DES crypte chaque bloc trois fois, chaque fois avec une clé différente. DES peut fonctionner dans plusieurs modes de bloc différents, y compris : le chaînage de blocs de chiffrement (CBC), le livre de codes électronique (ECB), le retour de chiffrement (CFB), le retour de sortie (OFB) et le mode de compteur (CTR). Chaque mode change la façon dont les fonctions de cryptage et la façon dont les erreurs sont traitées.

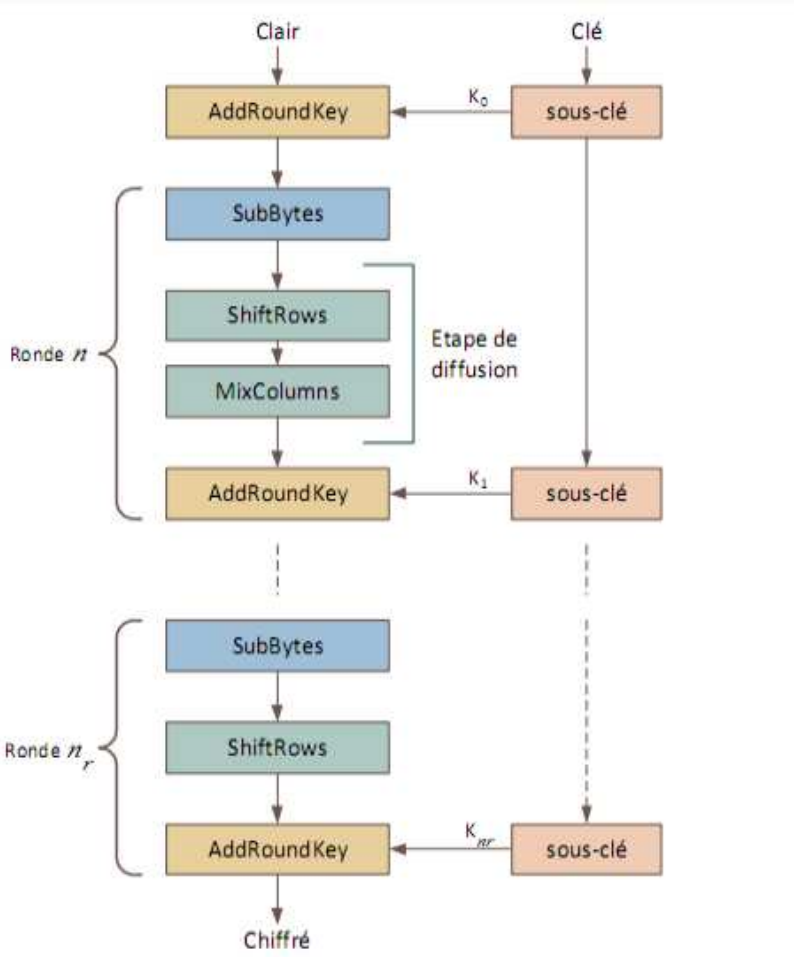

Figure 4. 2 Fonctionnement de l'algorithme AES [77]

B. Le Advanced Encryption Standard (AES)

AES est un ensemble de chiffrements de blocs symétriques approuvés par le gouvernement américain via le NIST. Il est maintenant utilisé par une variété de protocoles tels que SSH et IPsec. Il remplace le DES comme algorithme de chiffrement standard pour le gouvernement fédéral américain [77]. AES utilise trois chiffres différents : un avec une clé de 128 bits, un avec une clé de 192 bits et un avec une clé de 256 bits, tous ayant une longueur de bloc de 128 bits. C'un algorithme de chiffrement par bloc, pendant le processus de chiffrement/ déchiffrement il passe par 10 tours pour une longueur de clé de 128 bits, 12 tours pour 192 bits et 14 tours pour 256 bits. Le schéma présenté dans la figure 4.2 illustre le fonctionnement de l'algorithme AES.

Dans chaque tour, les transformations suivantes sont effectuées :

- La substitution des octets (Bute Substitution BS): transformé chaque octet du bloc à un autre octet en utilisant une fonction non-linéaire de substitution.
- La rotation des lignes (Shift Row SR): transposition des octets sur les lignes d'états AES. La première ligne reste intacte, les trois dernières lignes sont donc décalées.
- Le mélange des colonnes (Mix Column MC): corresponds à une multiplication matricielle à chaque colonne de l'état.
- L'addition des sous-clés (Add Subkey) : consiste à appliquer un XOR bit à bit entre l'état actuel et la clé correspondante au tour.

4.1.3.2. Les algorithmes asymétrique

Lorsque nous utilisons un algorithme asymétrique, une clé doit être distribuée d'une manière ou d'une autre. Il est nécessaire de communiquer la clé de manière hors bande, ce qui signifie que la clé ne doit pas être envoyée avec le message. Cela laisserait le message facilement accessible à un utilisateur indiscret. Lors de l'utilisation de la cryptographie à clé asymétrique, il n'est pas nécessaire de partager une seule clé. La clé publique est facilement disponible, et toute personne qui a besoin d'envoyer un message crypté n'a qu'a utilisé cette clé. En général, la sécurité des algorithmes asymétriques ne dépend pas de la faisabilité des attaques par force brute. Elle dépend plus de la faisabilité des opérations mathématiques inverses difficiles et des avancées de la théorie mathématique. Dans ce qui suit, nous examinons les deux algorithmes asymétriques Diffie-Hellman et RSA.

A. Diffie-Hellman (DH)

En 1976, après avoir exprimé leur désapprobation du DES et la difficulté de gérer les clés secrètes, Whitfield Diffie et Martin Hellman ont publié l'algorithme Diffie-Hellman (DH) pour l'échange de clés. Ce fut la première utilisation publiée de la cryptographie à clé publique, et sans doute l'une des plus grandes avancées du domaine de la cryptographie [78].

L'algorithme DH repose sur deux problèmes mathématiques. Le premier est le problème de logarithme discret (DLP), signifie l'application réciproque de l'exponentiation. C'est à dire, si on a un groupe fini « G » d'ordre « n » et $\beta \in G$, on doit trouver « a » tel que $0 < a < n-1$ de telle sorte que $g^a = \beta$.

Le deuxième est le problème Diffie-Hellman (DHP) qui est représenté d'une manière formelle comme suit : sachant que « a » et « b » sont des nombres entiers aléatoires, il est difficile de calculer « g^{ab} » même en connaissant g, g^a et g^b.

Donc le protocole d'échange DH basé sur DLP et DHP, vise à partager une clé secrète à travers un canal sécurisé. La description du protocole DH à clé publique est illustrée dans la figure 4.3. L'algorithme DH est utilisé par le protocole IPSec. IPSec utilise l'algorithme Diffie-Hellman en conjonction avec l'authentification RSA pour échanger une clé de session qui est utilisée pour chiffrer tous le trafic qui traverse le tunnel IPSec.

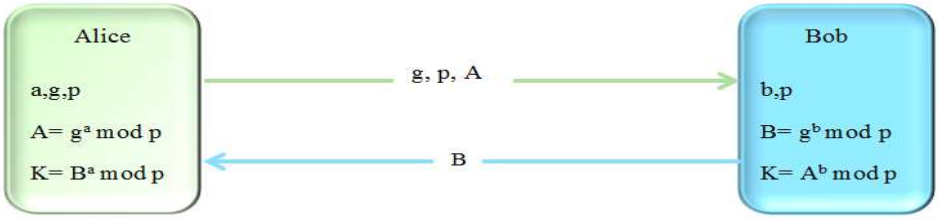

Figure 4. 3 Le protocole Diffie-Hellman

B. *Rivest, Shamir, Adleman (RSA)*

L'algorithme RSA, du nom de ses créateurs Ron Rivest, Adi Shamir et Leonard Adleman, est un algorithme asymétrique utilisé partout dans le monde, y compris dans le protocole SSL (Secure Sockets Layer) et IPsec. Il est utilisé pour sécuriser de nombreuses transactions courantes telles que le Web et le trafic de messagerie. RSA a été créé en 1977 et est toujours l'un des algorithmes les plus utilisés au monde [79].

La sécurité du protocole repose sur la difficulté de factoriser des produits des grands nombres entiers. La procédure de chiffrement RSA est exécutée en trois étapes.

1. Création des clés publique et privée, ou le récepteur effectue les opérations suivantes :

- Choix de p et q, deux nombres premiers distincts ;
- Calcule leur produit « n » (module de chiffrement) n = p∗q et φ(n) = (p - 1)(q - 1);
- Choix d'un entier naturel « e » (exposant de chiffrement). Tel que « e » est un entier premier avec φ(n) et strictement inférieur à φ(n);
- Calcule l'entier naturel « d » (exposant de déchiffrement), inverse de « e » modulo φ(n), et strictement inférieur à φ(n) ;

2. Chiffrement du message : Si M est un entier naturel strictement inférieur à n représentant un message, alors le message chiffré sera $C= M^e \pmod n$.
3. Déchiffrement du message : pour déchiffrer C, on utilise d, l'inverse de e modulo (p – 1)(q – 1), et l'on retrouve le message clair M par $M= C^d \pmod n$.

C. Les courbes elliptiques (ECC)

La cryptographie à courbe elliptique (ECC) peut être considérée comme une classe d'algorithmes cryptographiques Asymétriques [66]. Elle est nommée d'après le type de problème mathématique sur lequel reposent ses fonctions cryptographiques. ECC se base sur un problème impossible à résoudre mathématiquement. Les systèmes à courbe elliptiques ont été inventés indépendamment par Miler et Koblitz en 1985 [80]. ECC fournit un niveau de sécurité équivalent à celui des autres protocoles asymétriques (DH, RSA), mais avec des clés de taille réduite.

En utilisant ECC, les clés publiques et privées sont générées en fonction des paramètres de domaine de la courbe spécifique, nommé « E ». En plus de la courbe, un champ fini « F_p », un point de générateur ou point de base « G » et d'autres paramètres de domaine doivent être pré-convenus par les pairs communicants, Alice et Bob. Les clés privés sont des entiers choisis aléatoirement de l'intervalle [1.. n-1], où « n » est l'ordre premier du sous-groupe. Par la suite les clés privés sont générés à l'aide du point générateur « G » tel que Q_A/ q_A et Q_B/q_B sont les clés publiques/privées d'Alice et Bob respectivement.

$$Q_A = q_A * G \; et \; Q_A = q_B * G$$

Les deux parties vont échanger leurs clés publiques tout en gardant leurs clés secrètes en privé. En recevant les clés publiques, Alice va calculer « $q_A * Q_B$ » et Bob va calculer « $q_B * Q_A$ ». Par conséquent, ils obtiendront la même clé qui va être utilisé comme une clé symétrique pour le chiffrement et le déchiffrement des messages échangés. Ce processus est appelé le protocole d'échange de clé DH (ECDH).

La cryptographie basée sur ECC présente plusieurs avantages par rapport à d'autres types d'algorithmes. Le point de force de la ECC est qu'on peut utiliser des clés plus courtes, tout en conservant une forme de cryptage très sécurisée. C'est aussi un type d'algorithme très rapide et efficace. Les opérations ECC peuvent être implémentées sur du matériel. Donc, ECC peut être utilisée même dans des réseaux avec ressources limités (Puissance, mémoire, fréquence, bande passante…etc.) comme les WSNs et l'IoT. Ainsi, nous pouvons voir ECC implémenté dans une variété d'algorithmes cryptographiques, y compris l'algorithme de hachage sécurisé 2 (SHA-2) et l'algorithme de signature numérique à courbe elliptique (ECDSA).

D. Les signatures numériques (DS)

Les signatures numériques visent à signer un message afin de permettre la détection des modifications de son contenu. En outre elles permettent de s'assurer que le message a été légitimement envoyé par la partie attendue et empêchent l'expéditeur de nier qu'il a envoyé le message (non-répudiation). Ceci est possible grâce à la cryptographie asymétrique. Afin de signer numériquement un message, l'expéditeur génère un hachage du message, puis utilise sa clé privée pour crypter ce dernier, générant ainsi une signature numérique. L'expéditeur envoi par la suite cette signature avec le message, généralement en l'ajoutant au message lui-même. Lorsque le message est reçu, le destinataire utilise la clé publique de l'expéditeur pour décrypter la signature, rétablissant ainsi le hachage d'origine du message. Le récepteur peut ensuite vérifier l'intégrité du message en hachant à nouveau le message et en comparant les deux résultats. Le mécanisme de signature numérique est illustré dans la figure 4.4.

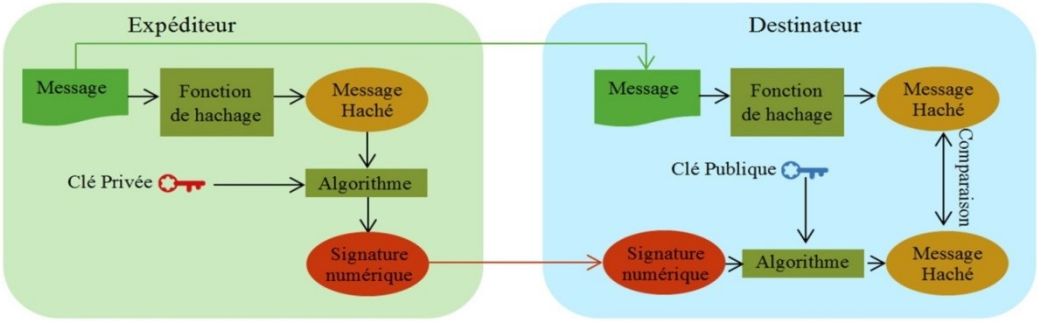

Figure 4. 4 Schéma de la signature numérique

Nous présentons ci-dessous deux exemple d'algorithmes de signature numérique. L'algorithme de signature RSA et l'algorithme de signature numérique à courbe elliptiques (ECDHA).

- **La signature RSA**

La signature RSA est un type de signature numérique qui utilise l'algorithme de clé asymétrique RSA présenté au-dessus. Le schéma de signature numérique RSA applique la clé privée de l'expéditeur à un message pour générer une signature. La signature peut ensuite être vérifiée en appliquant la clé publique correspondante au message. L'algorithme définit deux étapes essentielles ; la génération de la signature et la vérification de cette signature.

➢ Génération de la signature :

L'expéditeur ou l'initiateur calcul tout d'abord le hachage « h » du message a envoyé « M » en utilisant une fonction de hachage « H » tel que : h= H(M). Ensuite, il calcule la signature « S » en utilisant l'exposant « d » tel que $S= h^e \mod n$. A la fin l'expéditeur envois « M » et « S ».

➢ Vérification de la signature

Le destinateur calcul le hachage du message, h= H(M). Ensuite, il calcul « h' » à partir de la signature « S » et en utilisant l'exposant « e », tel que $h'= S^d \mod n$. A la fin, il compare h et h' afin de décider c'est le message est valide ou non.

- **Algorithme de signature numérique à courbe elliptique (ECDSA)**

L'ECDSA est un schéma standard de signature numérique à clé publique [81]. C'est un dérivé standardisé de l'ancien algorithme de signature numérique (DSA), conçu par David Kravitz [82]. Contrairement à DSA qui est basé sur un module arithmétique premier, l'ECDSA utilise des opérations à base des courbes elliptiques sur des champs finis. Par rapport à son prédécesseur, il a l'avantage d'être plus efficace. Il nécessite des longueurs de clé beaucoup plus courtes pour offrir le même niveau de sécurité. Le processus d'ECDSA se compose de trois étapes ; la génération des clés, la génération de signature et la vérification de la signature.

- Génération des clés:

L'expéditeur ou l'initiateur sélectionne un nombre aléatoire ou pseudo-aléatoire « d » dans l'intervalle [1.. n-1]. Ensuite il calcule « Q » tel que : $Q= d \cdot G$. Donc, la clé privée de l'expéditeur est « d » et sa clé publique est « Q ».

- Génération de la signature

Pour générer la signature (r, s) l'expéditeur effectue les opérations suivantes :

1: Choisir un nombre aléatoire « k » dans l'intervalle [1.. n-1] ;
2: Calcule les coordonnées $(x1, y2) = k \cdot G$;
3: Calcule $r = x1 \bmod n$; dans le cas ou r = 1 retourne à l'opération 1.
4: Calcule $k^{-1} \bmod n$;
5: Calcule e = H(M) ;
6: Calcule $s = k^{-1} (e+dr) \bmod n$; dans le cas ou s = 1 retourne à l'opération 1.

- Vérification de la signature

Pour vérifier la signature envoyée par l'expéditeur, le destinateur effectue les opérations suivantes :

1: Vérifie que « Q » est différent de « O » (le point à l'infini : $n \cdot Q = O$) et que « Q » appartient bien à la courbe elliptique
2: Vérifie que « s » et « r » sont des entiers dans l'intervalle [1.. n-1] ;

3: Calcule u1 = H(M) ⋅w et u2 = (r⋅ s^{-1} mod n) mod n ;

4: Calcule X= u1⋅G+ u2⋅Q

5: Calcule v = x1 mod n ;

6: Vérifie que r= v ; s'ils sont identiques donc la signature est acceptée.

En plus des hachages et des signatures numériques, il existe une autre construction qui peut intensifier l'utilisation de la signature de message, sous la forme de certificats numériques, communément appelés certificats [83]. Les certificats sont créés pour lier une clé publique à une entité en particulier et sont souvent utilisés comme forme d'identification électronique. Un certificat est généralement formé en prenant la clé publique et en identifiant des informations, telles qu'un nom et une adresse, et en les faisant signer par une autorité de certification (CA). Une autorité de certification est une entité de confiance qui gère les certificats numériques. Elle n'est qu'une petite partie de l'infrastructure qui peut être mise en place pour gérer des certificats à grande échelle. Cette infrastructure est connue sous le nom d'infrastructure à clé publique (PKI). Le certificat vise à vérifier qu'une clé publique est réellement associée à un individu particulier. Dans le cas de la signature numérique, il est possible que quelqu'un ait falsifié les clés utilisées pour signer le message et que les clés n'appartiennent pas réellement à l'expéditeur d'origine. Si nous avons un certificat numérique pour l'expéditeur, nous pouvons facilement vérifier auprès de l'autorité de certification pour nous assurer que la clé publique de l'expéditeur est légitime. Un certificat numérique peut être utilisé pour identifier un utilisateur, une machine ou un appareil avant d'accorder l'accès à une ressource, un réseau ou une application.

4.2. Mécanismes d'authentification d'IoT

L'IoT a besoin d'un mécanisme de vérification pour garantir que toutes les données sont envoyées et reçues uniquement par les parties concernées. Tous les appareils sont ouverts au piratage, dont il s'agit d'un capteur ou d'un appareil utilisé pour exécuter une fonction spécifique. Nous définirons dans cette section, l'authentification et les mécanismes proposés pour assurer cette propriété.

4.2.1. Authentification des systèmes IoT

Garantir la validité de l'identité d'un appareil accédant au réseau est la base de la sécurité. Les systèmes IoT se basent sur le mode de communication machine à machine. Pour de tels systèmes, l'authentification est très importante pour garantir la confidentialité et l'authenticité des données. L'authentification est un mécanisme par lequel un réseau détermine si l'utilisateur a accès à certaines ressources. L'authentification mutuelle est une exigence nécessaire pour immuniser le système contre l'usurpation des rôles des entités. Lorsque deux nœuds ou plus communiquent entre eux pour un objectif commun, ils doivent d'abord s'authentifier. Cette authentification doit se faire de manière mutuelle, sécurisée et efficace. Un appareil/ utilisateur doit pouvoir vérifier que les données reçues sont collectées par l'appareil/ utilisateur indiqué.

L'authentification traditionnelle dans l'IoT est principalement basée sur l'infrastructure à clé publique (PKI), qui nécessite une autorité centrale. Elle dépend d'une institution intermédiaire, c'est-à-dire un serveur CA. Une autre technologie utilisée pour garantir l'authentification IoT est les blockChain [84]. La technologie blockChain fournit des solutions réalisables pour l'authentification d'identité et la protection de la sécurité des dispositifs 'IoT.

4.2.2. Mécanismes d'authentification proposés pour les systèmes IoT

Les protocoles d'authentification mutuelle traditionnels sont souvent inefficaces pour les appareils à ressources limitées. En outre, les problèmes d'authentification et de contrôle d'accès dans l'IoT sont dus aux grands nombres d'appareils et à la nature de communication machine à machine. Plusieurs mécanismes ont été proposés pour résoudre le problème d'authentification d'appareils dans l'IoT [85-88]. La plupart de ces travaux utilisent des techniques cryptographiques à clé publique ainsi que des certificats numériques. Outre que les protocoles basés sur la clé publique, de nombreux protocoles d'authentification récents utilisent la cryptographie à clé symétrique [89]. Cette méthode

est appliquée pour implémenter à la fois, l'authentification mutuelle et le cryptage des données entre les extrémités d'un système utilisant la radio-identification (RFID)[2].

La cryptographie à courbe elliptique (ECC) est aussi une approche alternative à la cryptographie à clé publique. L'avantage des algorithmes cryptographiques basés sur ECC est clé à petite taille. Ils sont adaptés aux applications IoT. Par exemple, un algorithme de chiffrement ECC avec une taille de clé de 160 bits peut atteindre le même niveau de sécurité que celui du chiffrement RSA 1024 bits. En pratique, l'algorithme ECDSA et ECDH peuvent être utilisés dans des protocoles d'authentification mutuelle légers et offrent une sécurité de haut niveau.

Gaikwad et al. [90] ont utilisé l'authentification Kerberos pour les systèmes de maisons intelligentes. Ils ont utilisé l'algorithme de hachage sécurisé SHA1 et le standard AES pour la sécurité. Kerberos est un protocole d'authentification développé dans les années 80. Il permet à deux paires, auparavant inconnus l'une de l'autre, d'établir une clé secrète partagée grâce à l'utilisation d'un tiers de confiance. Les deux pairs partagent déjà une clé secrète avec ce tiers. Cependant, Kerberos n'est pas une solution durable pour l'authentification et AES n'est pas pratique pour les appareils IoT contraints.

Panwar et al. [91] ont proposé un mécanisme de sécurité pour l'IoT utilisant des certificats numériques avec le protocole DTLS. L'authentification est effectuée par des certificats numériques fournis par une autorité de certification. Ceci rend l'authentification plus robuste et remplace le mécanisme de clé pré-partagée dans DTLS. Cependant, l'utilisation des certificats n'est souvent pas viable pour les grands déploiements, tels que les réseaux 6LoWPANs.

Toujours en utilisant des certificats, Park et al. [92] ont proposé une structure de code d'autorisation, crée pour l'authentification des dispositifs IoT. Cette proposition simplifie la structure du certificat et convient aux dispositifs IoT. Le certificat existant est basé sur des signatures difficiles à appliquer sur les appareils IoT à ressources limitées, mais le code de confirmation est facile à gérer dans l'environnement IoT.

[2] Un Système RFID utilise la technologie d'identification à distance à l'aide de marqueurs et de lecteurs de radiofréquences.

Un autre protocole a été également proposé dans [93] pour les systèmes IoT distribués. Ce protocole est basé sur des certificats. Cependant, les informations d'identification cryptographiques sont stockées dans le nœud périphérique, exposant le protocole à des attaques de clonage.

Dans une autre partie, Zhao et al. [94] ont proposé un schéma d'authentification mutuelle asymétrique pour l'IoT. Dans leur schéma, l'authentification est effectuée entre le nœud terminal et la plate-forme. La fonction SHA1 et l'extraction de caractéristiques sont combinées dans le schéma proposé. Grâce à lesquels la sécurité de l'IoT est améliorée, les coûts de calcul et de communication sont également réduits.

Dans [95], les auteurs ont proposé un protocole d'authentification mutuelle léger. La proposition est basée sur un schéma de chiffrement à clé publique pour les applications des villes intelligentes. Ils ont proposé un nouveau schéma de chiffrement à clé publique. Le schéma et le protocole proposés conviennent aux appareils légers et aux réseaux à faible puissance déployés dans les applications IoT.

4.2.3. Comparaison entre les solutions proposées

L'analyse des protocoles / mécanismes d'authentification nous a conduit à identifier un certain nombre d'exigences et de problèmes ouverts à prendre en compte dans notre travail. Le tableau 4.1 présente une comparaison entre les mécanismes susmentionnés, proposés pour assurer l'authentification dans l'IoT.

Ref	Technique	Mécanisme de Sécurité	Limites
[90]	Kerberos	Cryptographie à clé symétrique	AES n'est pas recommandé pour l'IoT
[91]	Certificats numériques	DTLS (certificats numérique)	L'utilisation des certificats n'est pas viable pour l'IoT
[92]	Certificats simplifiés	cryptographie à clé publique	les signatures sont difficiles à appliquer sur les appareils IoT
[93]	Certificats	cryptographie à clé publique	Vulnérable aux attaques de clonage
[94]	ECC + SHA1	Cryptographie à clé	vulnérabilités héritées de

		symétrique (SHA1)	l'utilisation de SHA1
[95]	chiffrement à clé publique	cryptographie à clé publique	coût de calcul élevé pour les appareils IoT

Tableau 4. 1 Mécanismes d'authentification pour l'IoT

Parmi les exigences à prendre en compte, on doit trouver un compromis entre la consommation d'énergie et la sécurité. D'où les protocoles proposés pour les applications basées sur des capteurs, (contraints en termes de mémoire, de puissance de traitement, de batterie, etc.), doivent être légers, efficaces et sécurisés. En outre, la surcharge de communication des protocoles d'authentification est un facteur clé, en particulier lorsqu'il s'agit de périphériques à puissance limitée. Donc, le nombre de messages échangés entre les parties d'authentification doit être aussi bas que possible. Des faibles coûts de calcul doivent être appliqués lors de la conception de schémas d'authentification IoT. Donc, Il est nécessaire d'adopter des algorithmes et des protocoles cryptographiques légers lors de la conception de solutions d'authentification.

4.3. Solutions pour la gestion des clés dans L'IoT

La cryptographie est l'utilisation des codes et des chiffres pour protéger une communication et garder sa confidentialité de tout le monde sauf les destinataires prévus. Avec la croissance rapide de l'IoT et l'intégration de l'IP-based IoT, davantage d'appareils sont connectés à Internet, ce qui entraîne des échanges de données plus importants. Les contraintes exigées par des réseaux tels que les 6LoWPANs génèrent, à leurs tours, plus de risques de sécurité et de confidentialité, ce qui est actuellement un grand défi. Plusieurs approches ont été proposées pour garantir la création et l'échange efficace entre les appareils IoT [96]. La faisabilité des techniques de cryptographie à clé publique était très sceptique par rapport aux appareils à capteurs, les solutions proposées étant largement en faveur des techniques symétriques telles que la pré-distribution des clés [97]. De nos jours, les travaux plus récents [98] ont dûment évalué la viabilité des implémentations basées sur la cryptographie à courbe elliptique (ECC) même sur des appareils fortement contraints.

La gestion des clés est un problème de recherche important. Plusieurs solutions ont été proposées pour résoudre ce problème. Cependant, ces solutions sont applicables à d'autres systèmes en réseau, mais elles ne sont pas adaptées aux systèmes IoT. La gestion des clés dans le système IoT est un défi de recherche qui nécessite plus d'attention. Ceci constitue notre deuxième contribution dans cette thèse, qui est la proposition d'un protocole d'échange de clé pour l'IoT [J1]. Les systèmes de gestion des clés pour l'IoT peuvent être des solutions non-standards ou bien des solutions basées sur des protocoles Internet standards, comme DTLS, HIP et IPsec.

4.3.1. Solutions non-standards

Ils existent dans la littérature des solutions qui proposent des protocoles d'authentification et d'échange des clés pour l'IoT, indépendantes des protocoles Internet standard.

Un protocole léger basé sur la cryptographie symétrique est proposé dans [99]. Les clés générées sont utilisées dans le hachage du code d'authentification de message (HMAC). Les auteurs ont utilisé des clés de session à durée de vie limitée. Par conséquent, même si l'attaquant est capable de trouver une clé de session d'une manière ou d'une autre, cette clé de session (étant spécifique à cette session) ne serait pas utile pour décrypter les données des sessions futures. Le protocole proposé a deux clés partagées entre le dispositif IoT et la passerelle: la clé K_m, qui est une clé symétrique partagée à long terme et la clé K_s, qui est une clé de session symétrique partagée à court terme. En outre, K_iK_s est la clé de session initiale. Une mise à jour de la clé de session, Ks, a lieu périodiquement sur la base d'un nombre aléatoire et préalablement convenu. L'authentification est effectuée via des messages échangés entre les deux appareils, qui sont chiffrés par K_m et hachés par K_iK_s.

Une autre proposition qui profite de l'aspect hétérogène des appareils IoT a été proposée dans [100]. Les auteurs proposent une approche collaborative pour l'établissement de clés conçues pour réduire les exigences des protocoles de sécurité existants. Un appareil contraint peut déléguer sa lourde charge cryptographique à des nœuds moins contraints dans le même voisinage. Donc, ils exploitent un ensemble de nœuds intermédiaires tel

qu'un proxy, pour soutenir le processus d'établissement des clés. Une telle approche nécessite la disponibilité des dispositifs avec ressource complètes à proximité des dispositifs à ressources limitées. La sécurité du réseau dépend essentiellement de ce dispositif choisi. Cette solution n'est pas optimale car les deux nœuds d'extrémité peuvent ne pas avoir des liens de communication préétablis et sécurisée avec les proxys communs.

4.3.2. Solutions basées sur des protocoles Internet standards

Concernant les protocoles normalisés pour la sécurité des communications dans l'IoT, les protocoles DTLS, HIP et IPsec ont été proposé dans différentes recherches.

4.3.2.1. Solutions basées sur le protocole DTLS

Les communications basées sur le protocole DTLS établissent des clés entre les appareils communicants. L'établissement de clé nécessite des certificats X.509 et une infrastructure à clé publique. C'est une approche qui consomme souvent trop de ressources pour les appareils IoT. En outre, DTLS prend en charge l'utilisation de clés pré-partagées et les clés publiques brutes. Ces modes sont plus légers, mais ils ne sont pas évolutifs pour un grand nombre d'appareils.

Pour remédier à ça, Raza et al. ont proposé l'utilisation des clés symétriques pour DTLS [67]. Les auteurs ont proposé un mécanisme de sécurité négociable avec des clés symétriques. C'est une architecture pour l'IoT à ressources limitées, qui établit une relation de confiance entre les pairs de communication qui n'ont pas de relation directe préalable.

Un autre travail basé sur les clés symétriques est HIMMO [101]. Les auteurs ont proposé un système qui permet un partage de clés léger et basé sur l'identité. Ce partage se fait d'une manière non-interactive, afin que toute paire d'appareils connectés à Internet puisse communiquer en toute sécurité. Les auteurs affirment que leur schéma permet de sécuriser l'IoT en introduisant le mode de fonctionnement DTLS-HIMMO. Ce mode atteint les propriétés de sécurité de la suite de sécurité DTLS sans nécessiter des changements dans le protocole lui-même.

Au-delà des approches de sécurité basées sur la clé symétrique, les approches basées sur la clé publique sont de plus en plus pertinentes, telles que celles basées sur des certificats [72] ou des sur les identités [73]. Néanmoins, ces solutions ne sont pas évolutives pour L'IoT.

4.3.2.2. Solutions basées sur le protocole HIP

Une solution pour l'établissement de clés entre les nœuds à ressources limitées et les nœuds sans contraintes a été proposée dans [102]. Cette solution, appelée D-HIP. C'est un schéma d'échange de clés distribué pour l'IoT basé sur le protocole HIP. Les auteurs proposent un protocole d'échange de clés distribué et léger pour réduire les opérations cryptographiques coûteuses du protocole HIP. Les opérations cryptographiques lourdes sont déléguées à des nœuds moins contraints appelés proxys du voisinage. Ainsi, chaque proxy participe au calcul et à la livraison d'une partie de la clé DH publique. De plus, le proxy participe également à la dérivation de la clé DH commune avant sa construction par l'initiateur.

4.3.2.3. Solutions basées sur le protocole IPsec

Il existe un nombre croissant des recherches qui confirment l'efficacité d'IPsec pour les WSNs IP-connectés en particulier et pour l'IoT en générale. IPsec, étant un protocole Internet standard, est un choix optimal pour l'IoT, mais il nécessite certaines adaptations. Le protocole d'échange de clé par défaut d'IPsec, IKEv2, présente une limite pour l'utilisation sur des appareils avec contraintes. Afin de bénéficier des avantages multiples du protocole IPsec pour l'IoT (voir section 2.3.4, Chapitre II), plusieurs travaux ont proposé l'allègement d'IKE. Dans ce qui suit nous décrivons les travaux les plus importants pour la gestion des clés dans l'IoT, basés sur IKEv2 ou une version modifiée du protocole.

A. *Le protocole Lightweight IKEv2 pour IPsec compressé [103]*

Le travail de Raza et al. [103] est considérer comme le premier travail qui a démontré la faisabilité du protocole IKEv2 comme un protocole de gestion de clé pour IoT. Les auteurs ont proposé une implémentation légère d'IKEv2 visant à améliorer la gestion des clés pour les réseaux 6LoWPAN. Ils ont proposé des méthodes de réduction des en-têtes

pour rendre IKE plus adapté aux périphériques avec contraintes. Pour cela, ils ont introduit un schéma IKEv2 léger pour l'IPsec compressé. Ils ont défini un bloc de 4 bits pour la compression d'en-tête suivant, en anglais : Next Header Compression (NHC). Cet en-tête est reconnu par les bits d'identification "1101". Si l'indice de paramètres de sécurité (SPI) est égale à zéro, cela signifie que la SA définie par défaut est utilisée, au lieu d'une SA singulière avec un SPI égale à un, comme illustré dans la Figure 4.5.

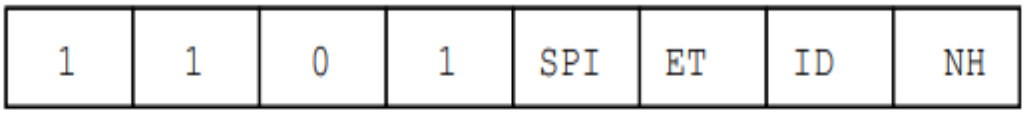

Figure 4. 5 Encodage NHC pour l'entête IKEv2 [103]

IKEv2 est un protocole qui établit une clé de session entre deux homologues. Alors que IKEv2 utilise la cryptographie asymétrique RSA, l'IKEv2 léger est basé sur ECC et le protocole Diffie-Hellmann pour l'échange de clés. Le but de ce travail est d'adapter IKEv2 aux environnements WSN IP-connectés, en utilisant la compression IPHC 6LoWPAN d'en-têtes IKE et des données de charge utile associées. Bien que les auteurs n'aient pas présenté d'analyse de performance à côté de leur proposition, ce travail est considéré comme référence pour les travaux postérieurs.

B. Le protocole IKEv2 modifié [104]

Ce travail présente une amélioration du protocole IKEv2 en termes de sécurité et de performances. L'amélioration consiste à utiliser la cryptographie ECC qui vise à atteindre un haut niveau de sécurité et d'efficacité. Le protocole proposé est composé de cinq messages d'échange (trois tours). Les trois premiers messages sont utilisés pour établir une IKE-SA où l'initiateur et le répondeur créent la IKE-SA, s'authentifient mutuellement et échangent leur clé secrète. Les deux derniers messages sont utilisés pour établir une IPSec-SA. Les auteurs proposent de minimiser également les opérations cryptographiques utilisées et d'utiliser à la place des calculs simples tels que les opérations XOR. Le schéma du protocole est présenté dans la figure 4.6.

Les auteurs affirment que leur protocole est plus efficace et entraîne moins de complexité de calcul que le protocole IKEv2 conventionnel, ce qui lui rend convenable pour les réseaux sans fil. Cependant, ce protocole est toujours lourd pour les appareils avec contraints et ne convient pas pour les systèmes IoT comme les 6LoWPANs.

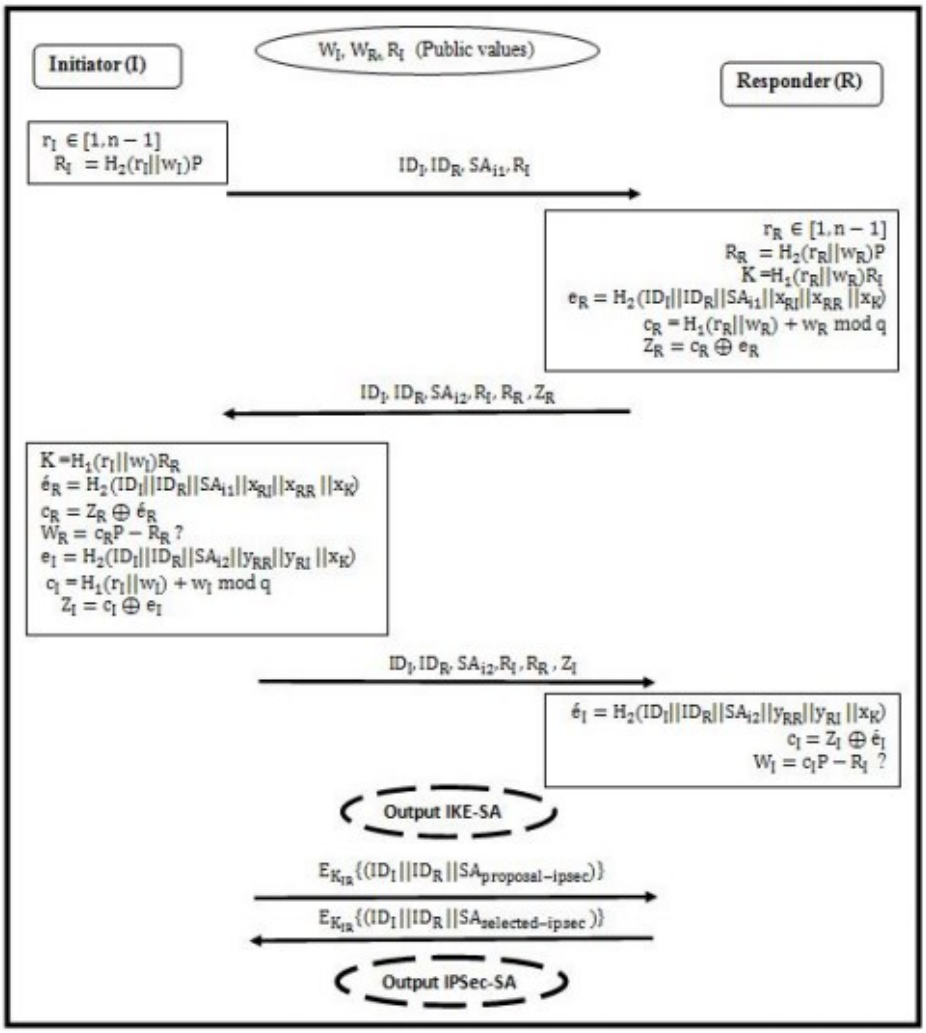

Figure 4. 6 Le protocole IKEv2 modifié [104]

C. *Le protocole LKA [105]*

Ce protocole est conçu pour fournir une sécurité de bout en bout entre les nœuds IPv6 et 6LoWPAN. Le protocole LKA est une adaptation d'IKEv2, c'est une implémentation

sans certificat. La valeur de hachage utilisée pour l'authentification ainsi que la vérification de la clé publique est calculée en utilisant le couple : adresse IP et identité des deux homologues. Ce protocole est basé sur Minimal IKEv2, une adaptation du protocole IKEv2 pour les nœuds contraints [106]. Le protocole Minimal IKEv2 ne prend pas en charge que la fin initiatrice du protocole. Ainsi, seuls les échanges initiaux IKE_SA_INIT et IKE_AUTH sont effectués. Ce protocole est utilisé pour les petits appareils faisant des communications de machine à machine. Dans de tels environnements, le nœud initiant les connexions est généralement très petit et l'autre extrémité du canal de communication est une sorte de périphérique plus grand.

Le protocole LKA proposé est décrit par les caractéristiques suivantes :

- Une seule suite cryptographique est utilisée ; il suffit donc de vérifier si SAi1 et SAr1 sont identiques.
- Élimine les certificats et les demandes de certificats et ECDH est utilisé à la place de l'échange de clés DH.
- La fonction de hachage utilisée est MBLAKE, une fonction de hachage unidirectionnelle basée sur une éponge, qui se révèle être plus efficace que les algorithmes traditionnels SHA-1 et LOCHA.
- Les échanges INFORMATIFS et les procédures de recomposition sont supprimés du protocole traditionnel IKEv2.

Concernant le fonctionnement du protocole, LKA est composé de 4 messages échangés entre l'initiateur et le répondeur. Dans les deux premiers messages, ils échangent des nonces et s'accordent sur les valeurs ECDH et les solutions cryptographiques. Dans les deux derniers messages, ils vérifient leur identité et calculent les clés secrètes. Après l'authentification mutuelle, l'initiateur et le répondeur conviennent de calculer les clés dérivées telles que la clé d'authentification, la clé de chiffrement et la clé d'intégrité à partir des éléments de la clé de session. Et puis les deux pairs communicants échangent les données en toute sécurité.

Bien que les auteurs proposent un protocole adapté à l'environnement IoT, leur proposition présente certaines lacunes, notamment l'attaque MITM. A savoir, un

attaquant peut se positionner entre l'initiateur et le répondeur pour espionner ou se faire passer pour l'un d'eux. Si cet intrus réussit, il pourra obtenir des valeurs utilisées pour calculer les clés secrètes. Sachant que ces clés sont utilisées dans les échanges suivants, l'emprunt d'identité augmentera le risque de révéler la clé de session par cet attaquant.

Selon leur proposition, l'initiateur choisit une clé privée di et envoie Pi = di * G au répondeur. Ainsi, le répondeur fait la même chose dans le message suivant et envoie son Pr = dr * G.

les échanges détaillée du protocole LKA sont présenté dans la figure 4.7.

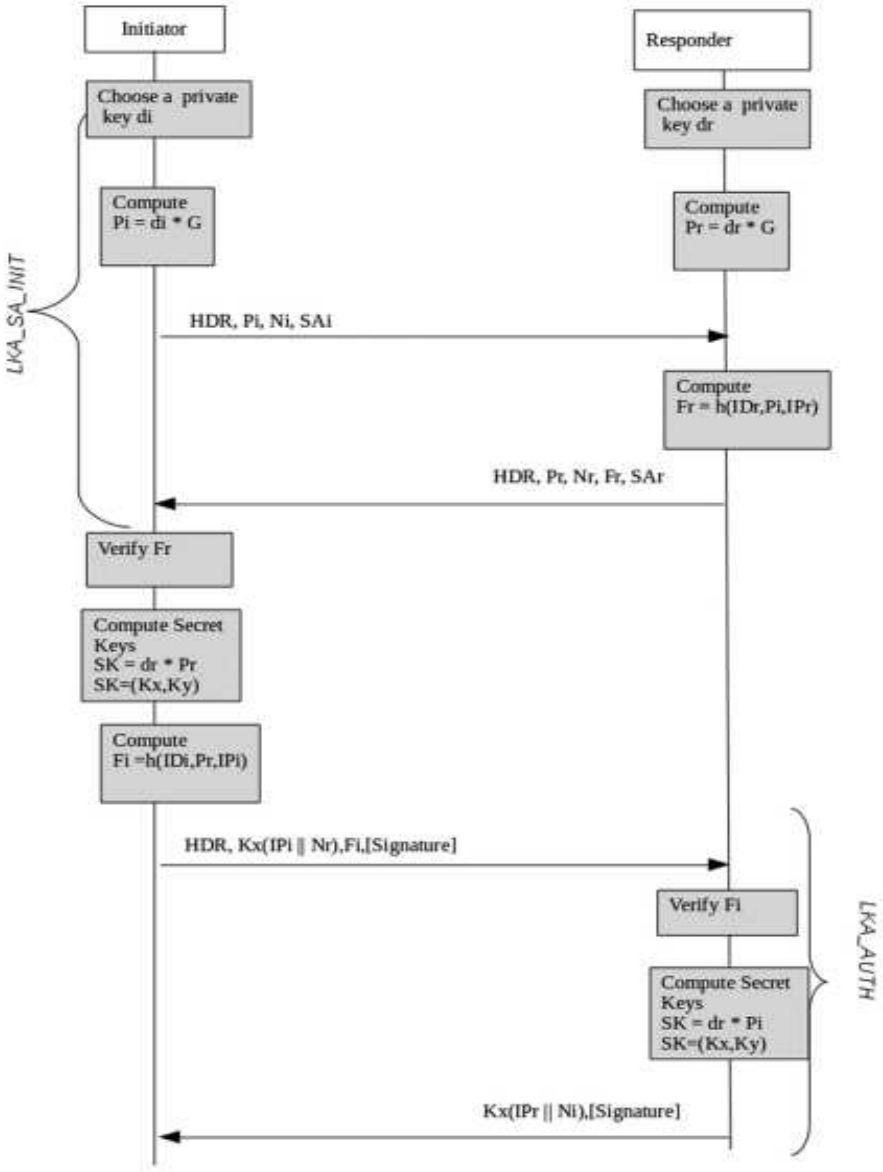

Figure 4. 7 les étapes détaillées du protocole LKA [15]

Si un attaquant "A" se positionne entre l'initiateur et le répondeur, afin d'écouter ou de se faire passer pour l'un d'eux, cet intrus pourra obtenir Pi ou Pr et les remplacer par son propre Pa = da * G, donnant l'impression qu'un échange normal est en cours. Notant que, Pi et Pr sont utilisés pour calculer les clés secrètes dans les échanges qui suivent, ce qui risque de révéler la clé de la session par l'attaquant.

D. *Le protocole TinyIKE [107]*

Raza et Magnusson [107] ont souligné qu'un accord est extrêmement essentiel dans l'environnement IoT à des fins de cyber sécurité. Ils ont présenté une adaptation légère d'IKEv2 pour le déploiement de l'IoT, appelée TinyIKE. TinyIKE s'appuie sur des certificats, des techniques cryptographiques à courbe elliptique, ainsi que des techniques cryptographiques symétriques.

Les auteurs supposent que la solution proposée résout le problème d'établissement des clés pour plusieurs protocoles IoT (RPL, IEEE 802.15.4) à l'aide d'une seule solution basée sur IKEv2. Avec l'aide de TinyIKE, ils ont pu résoudre le problème de gestion des clés pour plusieurs protocoles IoT en appliquant une seule approche basée sur IKEv2

Cependant, indépendamment de l'utilisation de l'ECDH, l'adaptation proposée n'a pas été accompagnée de modifications adaptées à l'environnement IoT. Au lieu de cela, les auteurs se sont concentrés sur l'utilisation de l'accélération matérielle qui se réfère à la mise en œuvre de protocoles cryptographiques dans le matériel à l'aide de capteurs embarqués, également appelés motes[3].

Concernant les solutions proposées dans la littérature, l'IKEv2 léger a un grand potentiel pour l'établissement des clés pour l'IoT, compte tenu de l'extensibilité d'IKEv2 et du fait qu'il est déjà utilisé pour la gestion des clés IPsec. Outre ses nombreux avantages, IKEv2 souffre de certaines lacunes de sécurité, telles que sa vulnérabilité contre les attaques DOS et MITM, en plus des coûts de calcul élevés causés par ses opérations compliquées. Par conséquent, un protocole d'authentification et d'échange de clé conçu particulièrement pour l'IoT est strictement nécessaire. Ceci nous a motivés à proposer dans ce travail, notre protocole d'échange de clé, TAKE-IoT [J1], pour qu'il soit utilisable avec IPsec et ainsi remplace le protocole IKEv2.

[3] est un nœud dans un réseau de capteurs qui est capable d'effectuer un traitement, de collecter des informations sensorielles et de communiquer avec d'autres nœuds connectés. Un mote est un nœud mais un nœud n'est pas toujours un mote [112]

4.4. Conclusion

Dans ce chapitre, nous avons discuté certaines notions de base de la cryptographie y compris : les clés cryptographique, la cryptographie symétrique et la cryptographie à clé publique. En outre, nous avons présenté les algorithmes cryptographique les plus connus comme : DES, RSA, DH. Ce chapitre présente également une vue d'ensemble sur les solutions proposée pour l'authentification des appareils IoT ainsi que les mécanismes d'échange de clés entre ces appareils. Nous avons présenté les travaux relatifs à l'authentification dans l'IoT, nous avons également fait une comparaison entre ces propositions. En outre, nous avons présenté les solutions de gestion et d'échange de clés, en se basant sur celles qui reposent sur les protocoles Internet standard comme DTLS, HIP et IPsec. En revanche, plus d'attention a été donnée aux travaux basés sur IPsec/IKE, ce qui sert le but de cette thèse.

Le but essentiel de ce travail est d'exploiter le protocole IPsec dans un environnement dynamique, comme l'IoT. Donc, ceci nous a motivés de proposé notre propre protocole d'authentification et d'échange de clé TAKE-IoT qui est présenté dans le chapitre VI. Cependant, avant de passer à la gestion des clés du protocole IPsec, nous commençons d'abord par les politiques de sécurité. Concernant la gestion automatique des politiques de sécurité IPsec, notre contribution est présentée dans le chapitre suivant.

CHAPITRE V

Gestion automatique des politiques de sécurité IPsec

Dans les chapitres précédents, nous avons étudié le protocole IPsec et la possibilité de l'utiliser pour sécuriser les communications de bout en bout dans des environnements dynamiques, comme l'IP-based IoT. Les politiques de sécurité sont un élément essentiel de l'architecture de sécurité du protocole IPsec. L'utilisation d'IPsec dans des environnements dynamiques nécessite une flexibilité dans le processus de la gestion des politiques. Ainsi, l'automatisation de ce processus est fortement recommandée. Dans ce chapitre, nous introduisons notre première contribution concernant le protocole IPsec. Notre travail se concrétise par la proposition d'un algorithme de génération automatique des politiques de sécurité IPsec sans conflits (AGCSP-IPsec) [119]. En outre, nous proposons notre méthode de détection des conflits et les mécanismes utilisés pour les résoudre [120].

Ce chapitre est composé de sept sections. La section 5.1 introduit le chapitre et les différentes parties de notre contribution. La section 5.2 présente des préliminaires sur les politiques de sécurité IPsec et leurs représentations. De plus, nous définirons les conflits et nous les classifions selon leurs types. La section 5.3 est réservée à notre mécanisme de gestion des politiques, ainsi que l'approche utilisée pour détecter les conflits et identifier leurs types. La section 5.4 explique la méthode utilisée pour la résolution des conflits. Dans la section 5.5, nous introduisons notre contribution principale qui est l'algorithme AGCSP-IPsec. L'évaluation des performances de l'algorithme proposé est présentée dans la section 5.6. En outre, nous présentons une comparaison entre notre proposition et les travaux relatifs. Finalement, nous concluons le chapitre dans la section 5.7.

5.1. Introduction

L'architecture de sécurité IPsec définit deux types de bases de données : la première est la base de données des politiques de sécurité (SPD) et la deuxième est la base de données des associations de sécurité (SAD). La base SPD contient des politiques de sécurité (SP) qui définissent les critères de contrôle d'accès. Le contrôle d'accès gère le flux des paquets IP entrants et sortants, en spécifiant le trafic qui doit être protégé avec IPsec de celui qui doit être contourné. La vérification des politiques de sécurité est un problème critique. Ce problème s'aggrave lors de l'utilisation d'IPsec dans des environnements dynamiques.

La nature de ces environnements et leurs contraintes présentes des défis majeurs pour la gestion efficace des politiques de sécurité. Dans la littérature, l'efficacité n'a pas été une préoccupation majeure dans les études précédentes. Ces études ont focalisé principalement sur les méthodes triviales de détection des conflits. Puisque ces conflits affectent le bon fonctionnement du protocole, nous proposons le protocole de génération automatique des politiques IPsec sans conflits, nommé AGCSP-IPsec.

Pour identifier les conflits, la SP IPsec est vérifiée dynamiquement, tout en conservant une complexité de calcul réduite. Après la résolution, une nouvelle politique sans conflit est générée. En plus de l'automatisation de la tache de vérification des SPs, notre proposition a pour but de minimiser, voir exclure l'intervention humaine nécessaire pour cette tâche. En outre, l'approche proposée vise à fournir : un service continu, une meilleure sécurité, un temps de traitement minimal et une complexité réduite.

La contribution présentée dans ce chapitre peut être divisée trois en parties :

- Premièrement, nous introduisons une classification des différents types de conflits qui peuvent exister dans une IPsec-SP. Ainsi, pour faire face à ces conflits, nous présentons un mécanisme de gestion de politique de sécurité. Le mécanisme proposé est basé sur la présentation des règles de la politique sous forme d'expression booléenne. La vérification des politiques se fait dynamiquement à l'aide d'opérations logiques au moyen du Diagramme De Décision Binaire Ordonnée (OBDD).
- Deuxièmement, nous proposons une méthode de résolution des conflits qui consiste à donner le privilège aux règles ayant la priorité la plus élevée. D'où le nom de la méthode proposée : préséance pour la règle la plus prioritaire (PHPR). En fait, PHPR utilise des données externes à côté des données essentielles (champs de condition et actions) ; ces données externes sont appelées « priorité ». Une priorité est un entier associé à chaque règle par l'administrateur réseau lors de la mise en œuvre de la politique. Cette méthode couvre la nécessité de l'intervention de l'administrateur pour résoudre les conflits, ce qui n'était pas garanti dans aucune des approches précédentes.

- Troisièmement, pour concrétiser notre travail, nous introduisons l'algorithme AGCSP-IPsec, un algorithme efficace pour la génération automatique de politique IPsec sans conflit. L'algorithme proposé est basé sur la manipulation des fonctions booléenne à l'aide d'OBDD et utilise la méthode PHPR pour la résolution de conflits. AGCSP-IPsec, vise à éliminer automatiquement les conflits d'une politique de sécurité, tout en minimisant le temps de traitement. En outre, une implémentation de l'algorithme est faite pour démontrer, d'une part, son efficacité et, d'autre part, ses avantages par rapport à d'autres algorithmes similaires.

Dans les sections suivantes, nous présentons chaque partie de notre contribution en détails.

5.2. Préliminaires

Dans ce qui suit, nous présentons des préliminaires, y compris les politiques de sécurité IPsec et leur représentation. De plus, nous introduisons notre classification des conflits tout en décrivant chaque type de conflit en détails. En outre, les notations utilisées dans ce chapitre sont présentées dans le tableau 5.1.

Symbole	Notation
P	politique de sécurité IPsec donnée
R_i	règle existante dans P
C_i	condition faisant partie de la règle R_i
a_i	Action associée à la règle R_i
S_i^n	Prédicat qui forme la condition C_i
A_f^a	ensemble formé par la conjonction des conditions qui déclenchent l'action a
F_{Total}	ensemble formé par l'union des fonctions d'agrégation
D^u	dispositif en amont (UpStream) dans le réseau
D^D	dispositif en aval (DownStream) dans le réseau
C_i^u	Condition d'une règle donnée dans la politique de dispositif en amont
C_j^D	Condition d'une règle donnée dans la politique de dispositif en aval

Tableau 5. 1 Notations utilisées pour la représentation de la politique de sécurité IPsec

5.2.1. Représentation de la politique de sécurité IPsec

IPsec vise à sécuriser la transmission d'informations sensibles sur des réseaux non-protégés. Les politiques de sécurité IPsec sont définies pour maintenir cet objectif.

Chaque SP regroupe une liste ordonnée de règles de filtrage définies par les administrateurs réseau. En effet, les politiques sont conservées dans des bases de données de politique de sécurité (SPD), où chaque implémentation IPsec contient un SPD nominal pour chaque interface et pour les deux directions par interface. La SP a pour but de déterminer le niveau de sécurité à appliquer sur le trafic Internet entrant et sortant.

Pour une SP IPsec donnée, le format générique utilisé dans la plupart des applications Internet définit l'ensemble des règles de filtrage des politiques comme une liste de contrôle d'accès (ACL). ACL est une liste de déclencheurs uniques, ce qui signifie que l'ordre des règles est très important. Chaque règle de la SP définit un filtre de paquets et une action. L'action sera exécutée sur tous les paquets qui correspondent au filtre associé. Les règles peuvent être présentées comme une liste ordonnée de prédicats sous la forme définie par l'équation (1). Sachant que, la condition C_i définit les prédicats (S) suivants: l'en-tête IP, le type de protocole, l'adresse IP source / destination et le numéro de port source / destination. En plus de ces prédicats standards, nous ajoutons le champ 'P' qui définit la priorité de chaque règle dans la politique.

$$R_i = C_i \rightarrow a_i \qquad (1)$$

Par conséquent, un paquet qui correspond à la condition C_i doit satisfaire ses prédicat et ainsi l'action sera appliquer sur ce paquet. Donc, C_i peut être formulé à l'aide de la fonction booléenne (2).

$$C_i = S_i^1 \wedge S_i^2 \wedge \ \ldots \wedge S_i^5 \qquad (2)$$

Pour chaque action, nous définirons une fonction d'agrégation A_f^a . Une fonction d'agrégation est la disjonction appliquée sur la conjonction de toutes les conditions de règles qui partagent la même action a_i comme défini l'équation (3). Prenant 'N' comme l'ensemble des règles qui définit a comme leur action associée. Donc, N= $\{i \mid R_i = C_i \longrightarrow a\}$ où $a \in \{Protect, Deny, Bypass\}$. Donc un paquet IP est soit protégé par IPsec (Protect), ignoré (Deny) ou bien passé sans protection (Bypass).

$$Af_a = \vee_{i \in N}(C_1 \wedge C_2 \wedge C_3 \wedge \ \ldots \wedge C_i) \qquad (3)$$

Enfin, nous définirons l'union des trois fonctions d'agrégation comme la fonction totale (F_{Total}) qui contient toutes les conditions d'une politique de sécurité donnée. Par conséquent, F_{Total} peut être défini par l'équation (4).

$$F_{Total} = Af_{Protect} \vee Af_{Bypass} \vee Af_{Deny} \qquad (4)$$

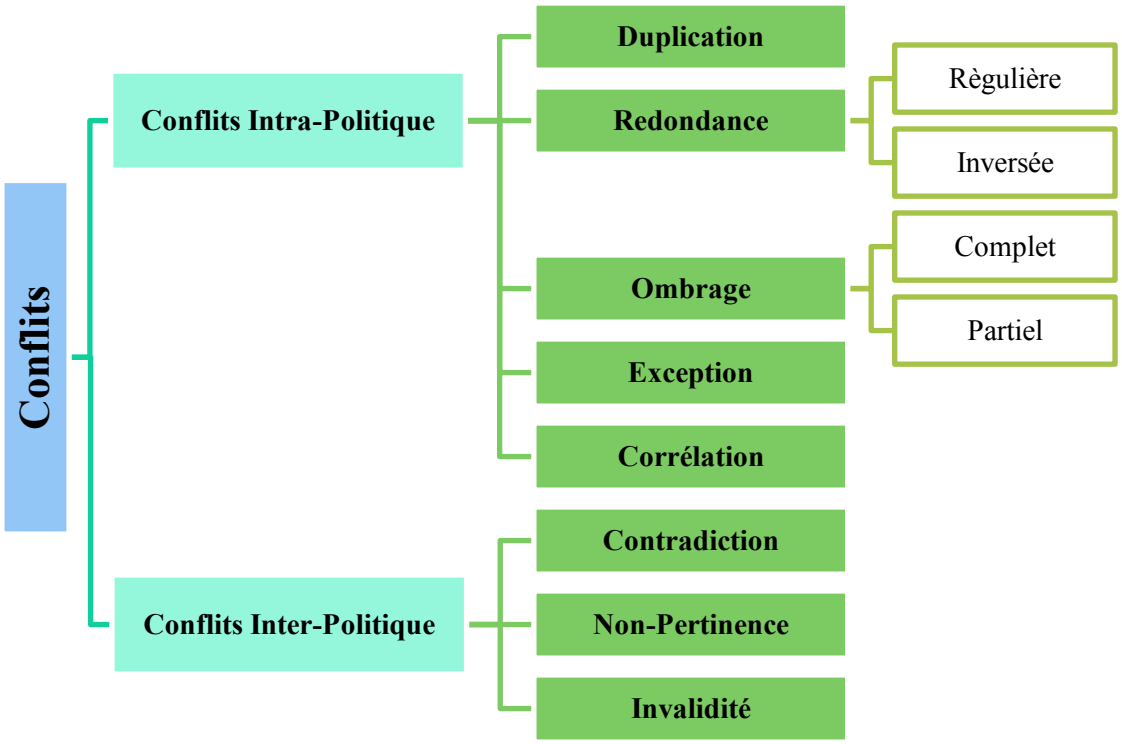

Figure 5. 1 Classification des conflits de politique de sécurité IPSec

5.2.2. Les conflits de la politique de sécurité IPsec

Les conflits d'une politique IPsec peuvent être divisés en deux catégories principales: les conflits Intra-Politique et les conflits Inter-Politique. Les conflits Intra-Politique sont des anomalies locales qui se produisent dans un même dispositif, c'est-à-dire dans la même politique. Ils sont généralement le résultat des mauvaises configurations ou des défauts humains. Sinon, les conflits Inter-Politique sont des anomalies causées par l'incohérence entre plusieurs politiques mises en œuvre dans des différents dispositifs. Dans ce travail, une classification des conflits est présentée comme le montre la figure 5.1.

Dans ce qui suit, nous prenons un exemple de deux règles : R_i et R_j. R_i est une règle existante dans la politique, notée la règle précédente. R_j est la nouvelle règle ajoutée

et notée la règle suivante. Les deux règles R_i et R_j sont définis selon (5) et (6) respectivement.

$$R_i = C_i \longrightarrow a_i \quad (5)$$

$$R_j = C_j \longrightarrow a_j \quad (6)$$

Selon la classification donnée, les conflits sont divisés en deux catégories principales, les conflits Intra-Politique (Intra-policy) et les conflits Inter-politique (Inter-Policy) comme suit :

5.2.2.1. Conflits Intra-Politique

Le processus de filtrage consiste à faire correspondre le trafic à toutes les règles de la politique jusqu'à ce qu'une correspondance soit trouvée. C'est-à-dire qu'il correspond à tous les champs de la condition d'une règle donnée. L'interdépendance entre deux règles quelconques dans la politique de sécurité peut être divisée en trois relations principales : règles disjointes (disjoint), identiques (matched) ou corrélées (correlated). Si deux règles ou plus sont identiques ou corrélées (dites aussi dépendantes), un conflit Intra-Politique est produit.

Le tableau 5.2 présente un exemple d'une politique de sécurité IPsec simplifiée. Nous avons construit cette politique pour faciliter la compréhension des conflits intra politique qui peuvent existés dans une politique de sécurité IPsec. Ainsi, l'exemple présenté dans le tableau contient une politique de 12 règles. L'interdépendance entre ces 12 règles produit 7 types de conflits qui sont : la duplication ; la redondance régulière et inversée, l'Ombrage complet et partiel, l'exception et la corrélation.

R#	Clause de la condition				Clause d'action		Type de conflit
	Source		Destina–tion		Action	paramètres	
	Adresse IP	N° Port	Adresse IP	N° Port			
R1	192.168.1.12	*	173.13.0.0	25	Protect	Protect ESP Tunnel to GW_B	Duplication avec règle R5
R2	192.168.2.0/24	*	173.13.1.0/24	*	Deny	**********	Redondance avec règle R4
R3	192.168.1.4	*	10.0.0.1	25	Protect	Protect ESP Tunnel to GW_C	Ombrage avec règle R7
R4	192.168.2.1	*	173.13.1.1	*	Deny	**********	Redondance avec règle R2
R5	192.168.1.12	*	173.13.0.0	25	Protect	Protect ESP Tunnel to GW_B	Duplication avec règle R1
R6	192.168.0.0/16	*	173.13.0.0/24	25	Protect	Protect ESP Tunnel to GW_B	Redondance avec règle R1
R7	192.168.1.4	*	10.0.0.1	25	Bypass	**********	Exception de la règle R8
R8	192.168.1.0/24	*	10.0.0.0/24	25	Protect	Protect ESP Tunnel to GW_C	Corrélation avec règle R7
R9	192.168.2.0	*	173.13.0.0	25	Bypass	**********	Exception de la règle R11
R10	192.168.2.2	*	10.0.0.0/24	25	Deny	**********	Corrélation avec règle R12
R11	192.168.2.0/24	*	173.13.0.0/24	25	Deny	**********	**********
R12	192.168.2.0/24	*	10.0.0.1	25	Protect	Protect ESP Tunnel to GW_C	Corrélation avec règle R10

Tableau 5. 2 Exemple simplifié d'une politique de sécurité IPsec

Dans ce qui suit nous définirons les types de conflits suivants :

- Duplication: Lorsque deux règles R_i et R_j sont exactement identiques, cela signifie que tous les prédicats des deux conditions sont identiques. En d'autres termes, R_i et R_j correspondent à la même classe de trafic et y appliquent la même action. Ainsi, R_i et R_j ont une relation d'appariement parfait et produisent un conflit de duplication défini par l'expression booléenne (7).

$$(i \neq j) \wedge (C_i = C_j) \wedge (a_i = a_j) \qquad (7)$$

- Redondance régulière (Regular Redundancy): Une règle est redondante lorsque tous les paquets qui satisfont à cette règle peuvent également satisfaire une ou plusieurs autres règles de la même politique avec la même action associée. Si R_i et R_j provoquent des conflits de redondance régulière, on peut dire que R_i et R_j ont une relation d'appariement de sous-ensemble. Le conflit de redondance régulière est défini par l'expression booléenne (8).

$$(i < j) \wedge (C_i \longrightarrow C_j) \wedge (a_i = a_j) \qquad (8)$$

- Redondance inversée (Inverted Redundancy): Ce conflit est similaire à la redondance régulière, sauf que la règle redondante est la règle précédente Ri. En d'autres termes, R_j est une généralisation de R_i. Ainsi R_i et R_j ont une relation d'appariement de super-ensemble. La redondance inversée est définie par l'expression booléenne (9). Il convient de mentionner qu'en raison du mécanisme de déclenchement unique dans les listes de contrôle d'accès, la règle dupliquée ou redondante ne sera jamais déclenchée. Par conséquent, ces conflits augmentent la taille de la politique de sécurité sans aucun avantage.

$$(i < j) \wedge (C_j \longrightarrow C_i) \wedge (a_j = a_i) \qquad (9)$$

- Ombrage complet (Complete Shadowing) : ce conflit est similaire au conflit de duplication, sauf que dans l'ombrage complet, les règles en conflit appliquent différentes actions au trafic commun. R_j est complètement ombrée par R_i, lorsqu'un trafic correspondant à R_i correspond également à R_j, et chaque règle applique une action différente sur ce trafic. Un conflit d'ombrage complet est défini par l'expression booléenne (10).

$$(i < i) \wedge (C_i = C_j) \wedge (a_i \neq a_j) \qquad (10)$$

- Ombrage partiel (Partial Shadonwing) : contrairement à un conflit d'ombrage complet, les règles qui causent ce type de conflit ne sont pas parfaitement identiques, et ne partagent qu'une partie du trafic commun. Par conséquent, si R_i et R_j cause un ombrage partiel, R_i est donc un sous-ensemble de R_j. Dans ce cas, les prédicats qui constituent leurs conditions ne sont pas tous identiques. R_i et R_j causent un conflit d'ombrage partiel si l'expression booléenne (11) est vraie.

$$(i < j) \wedge (C_i \longrightarrow C_j) \wedge (a_i \neq a_j) \qquad (11)$$

- Exception : les exceptions dans les politiques de sécurité ne sont pas toujours considérées comme des conflits. Dans certains cas, les exceptions sont faites intentionnellement par les administrateurs réseau pour des raisons de configuration. Ainsi, ces exceptions sont appelées exceptions intentionnelles. Dans notre modèle, pour faire la différence, les règles avec des exceptions intentionnelles ont toujours la priorité la plus élevée. R_i est une exception de R_j, si elles définissent une partie de trafic en commun, et si chaque règle effectue une action différente sur le trafic commun. Ainsi, R_j est un super-ensemble de R_i R_j. Le conflit d'exception est donné par l'expression booléenne (12).

$$(i < j) \wedge \left(C_j \longrightarrow C_i\right) \wedge (a_j \neq a_i) \qquad (12)$$

- Corrélation : dans une politique de sécurité, les règles peuvent être liées entre elles sans être un sous-ensemble ou un sur-ensemble. Dans ce cas, ces règles sont dites corrélées. Deux règles R_i et R_j sont en corrélation si R_i correspond à certains paquets qui correspondent au même temps à la règle R_j et vice versa. A noter que, les actions associées à ces deux règles sont distinctes. Une corrélation existe si l'expression booléenne (13) est vraie.

$$(i \neq j) \wedge \left(C_j \rightarrow C_i\right) \wedge (C_i \rightarrow C_j) \wedge \left(a_j \neq a_i\right) \qquad (13)$$

5.2.2.2. Conflits Inter-Politique

Ces conflits sont causés par des incohérences entre plusieurs politiques dans différents dispositifs IPsec. Les conflits Inter-Politique peuvent entraîner des menaces graves pour la sécurité, tels que : légitimer un trafic indésirable ou bloquer un trafic autorisé. Pour simplifier la compréhension du modèle, nous supposons avoir deux dispositifs IPsec différents : un dispositif en amont (Upstream) noté D^u et un dispositif en aval (DwonStream) noté D^d. Chacun de ces appareils possède sa propre politique de sécurité IPsec. Notre modèle définit trois types de conflits Inter-Politique basé sur les actions associées aux règles des deux dispositifs qui définissent le même trafic. Par conséquent, nous définissons trois type de conflits Inter-Politique : la contradiction, la non-pertinence et l'invalidité, comme montré dans le tableau 5.3.

Action du dispositif D^u	**Action du dispositif D^d**		
	Protect	**Bypass**	**Deny**
Protect	–	Non pertinence	Non pertinence
Bypass	–	–	Invalidité
Deny	Contradiction	Contradiction	–

Tableau 5. 3 Exemple de conflits Inter-Politique

Dans ce qui suit, nous donnons une définition détaillée de chaque type de conflits Inter-Politique.

- Contradiction : le conflit de contradiction se produit lorsque le dispositif en amont (Du) bloque un trafic autorisé ou protégé par la politique du dispositif en aval (D^d). R_i et R_j sont deux règles implémentées dans les politiques de sécurité des dispositifs D^u et D^d respectivement. R_i et R_j définissent la même classe du trafic. Ce trafic commun est permis ou protégé par R_j alors qu'il est ignoré (Deny) par R_i. Donc, si les deux règles correspondent parfaitement, alors c'est une contradiction totale. Cependant, s'ils sont appariés ou corrélés de manière inclusive, il s'agit d'un conflit de contradiction partielle. Un conflit de contradiction existe si l'expression booléenne (14) est vraie. Où C_i^u and C_j^d sont respectivement les conditions des règles des dispositifs D^u et D^d.

$$(C_i^u \wedge C_j^d) \wedge (a_i = deny) \wedge (a_j \neq deny) \qquad (14)$$

Le conflit de contradiction est une anomalie critique. La contradiction entre les règles des politiques appartenant à des différents dispositifs IPsec, échoue la négociation des associations de sécurité entre ces dispositifs. Par conséquent, le trafic sera abandonné au niveau du périphérique en amont, ce qui empêche un trafic légitime de circuler vers sa destination finale.

- Non-pertinence (Irrelevance) : le conflit de non-pertinence se produit lorsque le trafic est protégé par la politique de D^u, alors qu'il n'y a pas de règle dans la politique de D^d qui exige la protection de ce trafic. En d'autres termes, R_i et R_j partagent un trafic commun, mais R_i protège ce trafic tandis qu'il est ignoré par R_j. Le conflit de non-pertinence est défini par l'expression booléenne (15).

$$(C_i^u \wedge C_j^d) \wedge (a_i = Protect) \wedge (a_j \neq Deny) \quad (15)$$

- Invalidité (Spuriousness): ce conflit se produit lorsque la politique du dispositif D^u autorise un trafic ignoré par la politique D^d. Par conséquent, si R_i et R_j partagent un trafic commun et R_i permet ce trafic alors qu'il est rejeté par R_j, alors les deux règles provoquent un conflit d'invalidité. Les conflits d'invalidité augmentent les vulnérabilités du réseau, car ils permettent au trafic indésirable de circuler dans le réseau. A noter que, tous les dispositifs intermédiaires doivent contourner tout trafic autorisé par le dispositif source. Donc, Ce conflit se produit si l'expression booléenne (17) est vraie.

$$(C_i^u \wedge C_j^d) \wedge (a_i = bypass) \wedge (a_j = deny) \quad (17)$$

5.3. Mécanisme de gestion des politiques de sécurité IPsec

Après avoir démontré la représentation de la SP IPsec en utilisant un modèle d'équations Booléenne, dans cette sous-section, nous présentons notre mécanisme de gestion des SP IPsec, ainsi que l'approche utilisée pour détecter les conflits et identifier leurs types.

5.3.1. Vérification de la politique

Cette étape commence par l'exemption des règles non-conflictuelles[1]. Ainsi, ces règles ne contribuent pas dans les phases de détection et de résolution. Pour chaque nouvelle règle ajoutée notée R_{new} sa condition C_{new} est comparée avec F_{Total}. Donc, R_{new} est jugée non conflictuelle si l'équation (18) est vraie.

$$F_{Total} \wedge C_{new} = false \qquad (18)$$

Si l'équation (18) est vraie, R_{new} est ajoutée automatiquement à une nouvelle politique de sécurité sans-conflit (Conflict-Free Policy). Par conséquent, sa condition est ajoutée à la fonction d'agrégation correspondante à son action associée (a) tel que défini dans l'équation (19).

$$Af_a = Af_a \vee C_{new} \qquad (19)$$

Dans le cas contraire, si l'équation (18) n'est pas vérifiée, cela signifie que R_{new} a une interdépendance avec au moins une autre règle au sein de la même politique. R_{new} est donc considérée comme une règle conflictuelle.

Pour l'analyse Inter-politique, le principe est le même. Cependant, R_{new} est comparé à toutes les fonctions d'agrégation (correspondantes à l'action associée à R_{new}) de toutes les politiques appartenant aux périphériques du même chemin (Same-Path Device)[2]. A savoir, P' est une politique implémenté dans un périphérique du même chemin, P' définit la fonction totale F'_{Total}. Ainsi, un conflit Inter-politique est provoqué par R_{new} si l'expression Booléenne (20) est valide.

$$(F'_{Total} \wedge C_{new}) \wedge !(Af'_a \wedge C_{new}) \qquad (20)$$

La première partie de cette expression indique que R_{new} existe également en P '. Par conséquent, R_{new} est considéré comme une règle pertinente notée R'[3], ce qui signifie que R' définit la même classe de trafic de R, et donc ils partagent des prédicats communs. Cependant, la deuxième partie indique que R et R' sont associés à des actions différentes.

[1] Règle non-conflictuelle: une règle indépendante qui est dissociée des autres règles de la politique de sécurité, ou son association ne provoque pas de conflits avec d'autres règles.
[2] Périphérique de même chemin: périphérique IPsec qui contribue au routage d'un paquet de sa source à sa destination finale dans le même sous-réseau.
[3] Règle pertinente: règle qui a une interdépendance avec une autre règle de la même politique ou d'une politique d'un autre périphérique de même chemin.

Concernant le processus de détection des conflits, nous basons notre méthode sur la relation entre les règles pour détecter le type de conflit. Si la condition R_{new} est intersectée avec F_{Total} comme le montre l'expression (21).

$$C_{new} \Longrightarrow F_{Total} = true \qquad (21)$$

Cela signifie que R_{new} est dépendante et qu'elle peut donc avoir l'une des quatre relations suivantes avec d'autres règles :

1. Match parfait : lorsque deux règles partagent exactement les mêmes prédicats de conditions.
2. Sous-assemblage : lorsque les prédicats de la nouvelle règle composent un sous-ensemble d'une règle existante dans la même politique.
3. Généralisation : lorsque les prédicats de la nouvelle règle englobent des prédicats d'une règle existante dans la même politique.
4. Corrélation : lorsqu'une partie des prédicats de la nouvelle règle compose un sous-ensemble d'une règle existante et une autre partie est une généralisation de cette règle existante.

Par souci de compréhension, nous considérons le tableau 5.4 qui présente un exemple simplifié de chaque relation définie ci-dessus. Ainsi, la partie condition comporte deux champs : l'adresse IP source présentée en 2 bits (Src) et pareil pour l'adresse destination (Dst). En outre, le caractère générique (*) dans un champ signifie que la partie condition de la règle n'a pas de contraintes particulière.

Règle	Condition		Action	Relation
	Src	Dst		
1	00	*	Deny	Match parfait
2	00	*	Deny	
3	01	11	Bypass	Sous-assemblage
4	01	*	Protect	
5	*	10	Protect	Généralisation
6	01	10	Deny	
7	10	*	Protect	Corrélation
8	*	11	Bypass	

Tableau 5. 4 Les relations existantes entre les règles d'une politique de sécurité IPsec

Dans ce travail, les conflits intra-politiques sont systématiquement classés selon les relations prédéfinies. Ainsi, une nouvelle catégorisation des conflits intra-politiques selon leurs actions, est proposée comme le montre la figure 5.2.

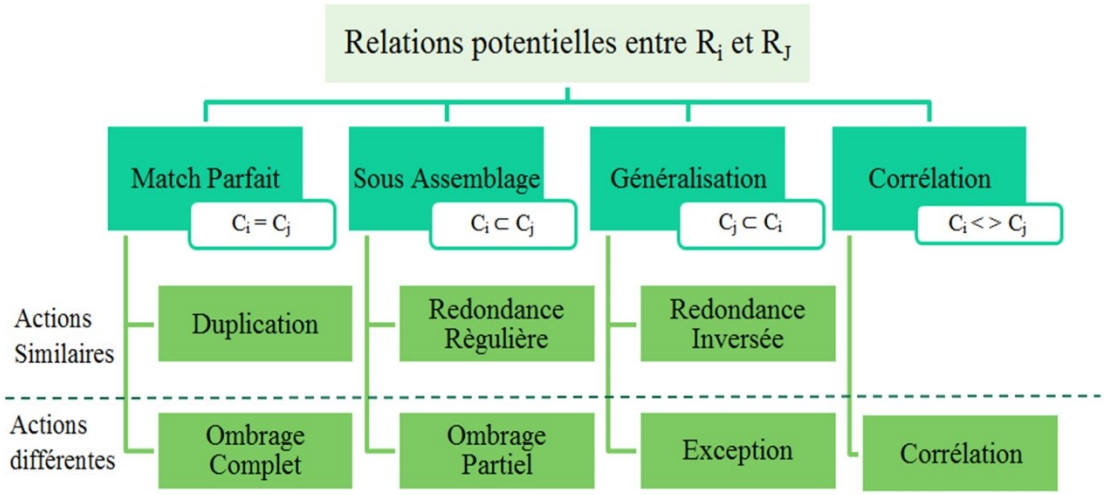

Figure 5. 2 Catégorisation proposée des conflits Intra-Politique

5.3.2. Détection et identification des conflits

Indépendamment de la relation existante entre deux règles d'une politique de sécurité, il existe deux cas différents : soit les règles partagent les mêmes actions associées, soit elles appliquent différents actions sur le trafic en commun. Partant de ce concept, nous avons exploité cette caractéristique dans le processus de la détection des conflits. Le principe est simple, il suffit de vérifier la relation entre C_{new} et la fonction d'agrégation A_f correspondante à l'action de la nouvelle règle. Si (22) est vérifiée, cela signifie que les deux règles partagent les mêmes actions. Donc le type de conflits est soit : duplication, redondance régulière ou bien redondance inversée. Dans le cas contraire, le conflit est : un ombrage total ou partiel, une exception ou bien, une corrélation. Ces quatre types de conflits sont plus compliqués, car toutes les règles impliquées peuvent affecter la cohérence de la politique. Par conséquent, l'élimination de ces règles ne peut pas se faire automatiquement. Plus de détails seront donnés dans la section suivante.

$$Af_a \wedge C_{new} = false \quad (22)$$

Pour la détection des conflits Inter-Politique, le principe est le même. Seulement, la condition de la règle est comparée aux politiques des autres périphériques du même chemin. En fait, un conflit Inter-Politique se produit lorsque deux règles dans deux SP différentes définissent la même classe de trafic, alors que ces règles n'effectuent pas le même traitement sur ce trafic. Ainsi, la clé pour détecter un conflit Inter-Politique est de vérifier les actions des règles en conflit.

La proposition de cette technique a fait notre première contribution qui a été acceptée et présentée à la conférence internationale «Seminar on Detection Systems Architectures and Technologies (DAT 2017) », ensuite publiée dans la bibliothèque digitale du IEEE [120].

5.4. PHPR : La méthode proposée pour la résolution des conflits

Une fois qu'un conflit est détecté et que son type est identifié, l'étape suivante consiste à résoudre l'anomalie causée par ce conflit sans compromettre la sécurité de la politique. La résolution des conflits de politique de sécurité IPsec n'a pas reçu beaucoup d'attention dans les approches précédentes, la principale préoccupation de ces travaux était d'identifier le type de conflit sans aucun mécanisme de reprise. Ainsi, nous proposons dans ce travail différentes techniques de résolution, telles que: l'élimination des règles, la réorganisation des règles à l'aide de la technique PHPR et la modification des règles.

Les conflits qui n'affectent pas la cohérence de la politique (la duplication, la redondance régulière et la redondance inversée) sont dits conflits Auto-Résolus (Auto-resolved conflicts). Nous avons choisi cette notation puisque l'élimination des règles impliquées dans ce type de conflits est inoffensive pour la SP. Par conséquence, une fois qu'un conflit Auto-Résolus est détecté, le processus de résolution est déclenché et les règles en conflit sont automatiquement éliminées de la SP.

Mis à part les conflits Auto-Résolus, l'élimination des règles n'est pas la solution optimale, car elle peut entraîner des incohérences dans la politique et menacer la sécurité du réseau dans son ensemble. À cette fin, la technique de réorganisation des

règles est utilisée dans notre approche. Tirant parti de l'aspect Déclencheur-Unique (Single-Trigger) des ACLs, la technique de la première règle de correspondance (FMR) est utilisée pour résoudre les conflits dans notre modèle proposé. FMR est basé sur la sélection de la première règle applicable dans une liste ordonnée de règles de filtrage. Il y a plusieurs travaux utilisés dans ce domaine, comme dans [108] où les auteurs ont proposé la technique nommée : priorité pour la règle la plus spécifique (MSTP). Cette technique est utilisée pour les politiques des Pare-Feu, elle impose l'application de l'action de règle avec les prédicats les plus spécifiques.

Inspirés par ces stratégies, nous proposons une nouvelle technique pour la résolution des conflits de politique de sécurité IPsec : préséance pour la règle la plus prioritaire, en anglais : Preference for the Highest Property Rule (PHPR). Cette technique utilise des données supplémentaires à côté des données essentielles (clauses de condition et d'action). Ces données supplémentaires sont dites : priorités. La priorité est associée à chaque règle de la politique afin de résoudre les conflits, en particulier dans le cas de règles à conflits multiples ; où des conflits se produisent entre plus de deux règles. Dans notre proposition, nous supposons que la priorité de la règle est un entier associé à chaque règle par l'administrateur réseau lors de la mise en œuvre de la politique. Sachant que la priorité n'a aucun impact sur le filtrage des paquets, elle n'est utilisée qu'à des fins de résolution de conflits. Pour générer des règles prioritaires. Les administrateurs doivent associer un entier à chaque règle lors de l'implémentation des contraintes de la SP. Cet entier varie de 1 à 5, où 1 est la priorité la plus faible et 5 est la priorité la plus élevée. Ainsi, la priorité la plus élevée n'est utilisée que pour marquer les règles avec des exceptions intentionnées[4].

Il est à noter que PHPR peut être utilisée aussi pour modifier une règle. Dans le cas d'un conflit d'ombrage partiel, l'action d'une règle peut être remplacée par l'action de la règle la plus prioritaire. Plus de détails sont donnés dans ce qui suit.

5.5. Notre algorithme AGCSP-IPsec proposé

Afin de résoudre les problèmes de conflit susmentionnés, nous présentons notre algorithme proposé pour la génération automatique d'une politique de sécurité IPsec

[4] Les exceptions intentionnées ne sont pas des conflits puisqu'elles sont faites par les administrateurs pour des raisons de configuration ou pour résoudre des conflits d'autre type (voir sous-section 5.1.4.3).

sans conflit, en anglais : Automatic Generation of Conflict-free Security Policy of IPsec (AGCSP-IPsec). L'organigramme global de l'algorithme proposé est illustré dans la figure 5.3.

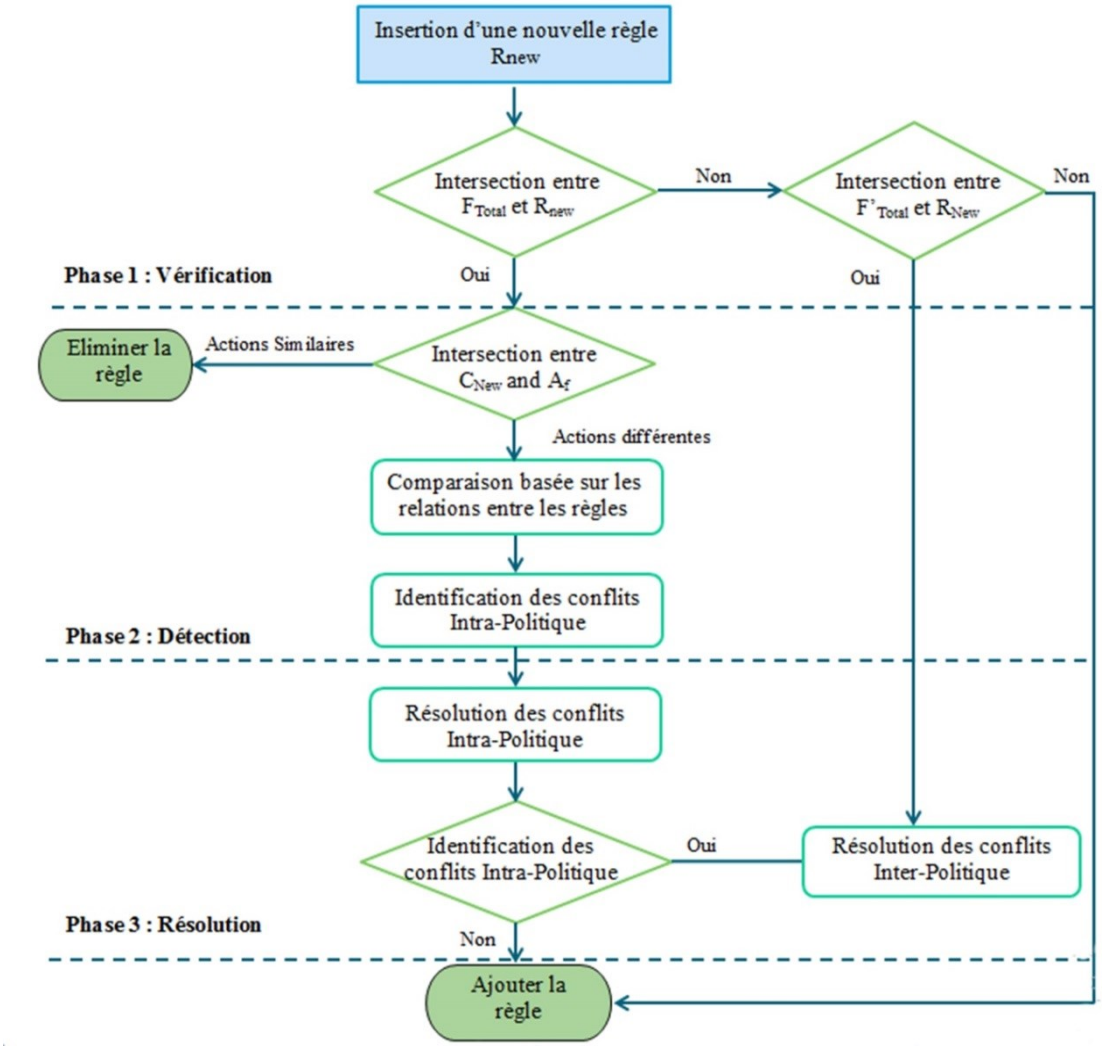

Figure 5. 3 L'organigramme global de l'algorithme AGCSP-IPsec

L'algorithme AGCSP-IPsec comporte trois phases principales :

1. Première phase : vérifie dynamiquement la politique de sécurité où les règles non-conflictuelles sont ajoutées automatiquement à la politique sans passer aux deuxième et troisième phases.
2. Deuxième phase : découvre le type de conflit provoqué par les règles conflictuelles résultantes de la première phase.
3. Troisième phase : résolution des conflits détectés dans la deuxième phase, selon leurs types.

Dans ce qui suit, nous décrivant en détails le fonctionnement de l'algorithme AGCSP-IPsec dans chacune de ces phases (voir l'algorithme 5.1).

Algorithm 5.1. AGCSP-IPsec main algorithm

Input: IPsec security policy

Output: Conflict- free IPsec security policy

```
1:Begin
2:   Policy: P1, P2, P'; Rule: R
3:   R= Get_Rule (P1);
4:   while (R) do
5:       if (empty (P2)) ||! (R.cond ∧ P2.Total_Fun) &&
         ! (P1.Agg_Fun [R.action] ∧ R.cond) then
6:               Add_Rule (R, P2);
7:               P2.Agg_Fun [R.action] = P2.Agg_Fun [R.action] ∨ R.cond;
8:               Refresh (P2.Total_Fun);
9:       else if (P1.Agg_Fun [R.action] ∧ R.cond ) then
10:              P2= Intra_Conflicts_Detetcion (P1, R); //Algorithme 5.2
11:      end if
12:  end while
13: return Inter_Conflicts_Detetcion (P2, P'); // Algorithme 5.3
14:end
```

Algorithme 5. 1 l'algorithme principal de l'AGCSP-IPsec

5.5.1. Phase 1 : Vérification dynamique de la politique

La vérification de la politique vise à juger si une règle causera ou non un conflit, lors de son ajout. Les règles non-conflictuelles sont exemptées du processus de détection. L'importance de la variable booléenne F_{Total} est distinguée dans la phase de vérification. La comparaison des conditions des règles avec la fonction F_{Total} couvre la nécessité de faire une analyse itérative de toutes les règles, chaque fois qu'une nouvelle règle est ajoutée. Sachant qu'après une analyse itérative, les résultats peuvent montrer que la règle est indépendante. Dans ce cas, l'analyse de la politique peut prendre du temps sans avantages. Par conséquent, la fonction F_{Total} est la solution alternative pour une vérification dynamique des politiques de sécurité.

Dans cette phase, OBDD est utilisé pour représenter les règles. Un OBDD est construit à l'aide des fonctions booléennes composées à partir des conditions associées aux règles de la SP. La partie action est utilisée pour choisir la fonction d'agrégation correspondante. Dans le but de montrer comment l'algorithme AGCSP-IPsec fonctionne dans la première phase, nous présentons les étapes essentielles pour cette phase.

- Étape 1: en utilisant OBDD, l'algorithme AGCSP-IPsec transforme la partie condition de R_{new} sous une forme binaire. Dans le cas où la règle R_{new} est la première règle à ajouter à la SP sans conflit, l'ajout de la règle se fait automatiquement en haut de la SP. Lorsque R_{new} est ajouté à la politique sans conflit (Algorithme 5.1 : ligne 5), la condition est placée dans la fonction d'agrégation correspondante en fonction de l'action associée à cette règle (algorithme 5.1: ligne 6). Ainsi, la fonction F_{Total} est automatiquement mis à jour (Algorithme 5.1 : ligne 7).
- Étape 2: l'algorithme vérifie si l'ajout de R_{new} provoque des conflits Intra ou Inter politique (Algorithme 5.1 : ligne 9). La vérification des conflits Inter-Politique est effectuée en comparant la condition de R_{new} avec toutes les fonctions F'_{Total} des politiques implémentée dans les périphériques du même chemin. Si aucun conflit n'est détecté, la règle R_{new} est ajoutée directement à la SP (23).

$$Af_a = Af_a \vee C_{new} \quad (23)$$

- Étape 3: le résultat affirmatif de la deuxième étape, prouve que l'ajout de la règle R_{new} va provoquer un conflit dans la politique (24).. Donc, la deuxième phase sera déclenchée automatiquement.

$$C_{new} \Longrightarrow F_{Total} = true \quad (24)$$

5.5.2. Phase 2 : identifier le type de conflit

Le but de cette phase est de découvrir le type de conflit provoqué par les règles conflictuelles résultantes de la première phase. Ceci est accompli selon le schéma de classification proposé dans la sous-section 5.2.2.

Algorithm 5.2. Intra-Policy conflict detection function

```
1:Begin
2:   Policy: P2; Rule: R, R2;
3:   if (R.cond ⋀ P.Agg_Fun [R.action]) then
4:        Drop_Rule(R);
5:        Alert ("Rule Dropped");
6:   else
6:        R2=find_Conflicting_Rule (R);
7:        while (R2) do
8:          if (R.cond ⋀ R2.cond) then
9:             P2= Complete_Shadowing_resolution(R, R2)
10:        else if (R.cond ⋀ R2.cond == R1.cond) then
11:                P2= Partial_Shadowing_resolution(R, R2)
12:             else if (R.cond ⋀ R2.cond == R2.cond) then
13:                   P2= Exeception_resolution(R, R2)
14:                else P2= Correlation_resolution (R, R2)
15:                end if
16:         end if
17:       end while
18:   end if
19:  Return P2;
14:end
```

Algorithme 5. 2 Pseudocode de la fonction de détection des conflits Intra-Politique

Le processus de détection des conflits Intra-Politique se déroule en trois étapes qui sont décrites ci-dessous.

- Étape 1 : R_{new} qui a été identifiée comme une règle conflictuelle dans la première phase, est d'abord comparée à la fonction d'agrégation qui correspond à son action associée (Algorithme 5.2 : ligne 2). Si le résultat de la comparaison est affirmatif, cela signifie que l'ajout de R_{new} va provoquer un conflit avec une ou plusieurs règles de la SP. Cependant, ces règles appliquent les mêmes actions sur le trafic en commun avec Rnew. Donc, R_{new} est automatiquement éliminée, car la suppression de la règle n'a aucun impact sur la cohérence de la politique (Algorithme 5.2 : ligne 3).

- Étape 2 : dans le cas contraire, si la comparaison dans la première étape est négative alors R_{new} et les règles pertinentes ne partagent pas la même action. Dans ce cas, un OBDD de toutes les règles pertinentes est construit afin de découvrir les règles impliquées (Algorithme 5.2 : ligne 6).
- Étape 3: le type de conflit est détecté en fonction de la relation entre Rnew et les règles pertinentes. Par exemple, nous supposons que R_{new} provoque un conflit avec une règle pertinente notée R_{rev}. Donc, R_{new} et R_{rev} sont essentiellement interdépendants, et l'expression (25) est vraie.

$$C_{new} \wedge C_{pre} \neq false \quad (25)$$

Lorsque l'expression (25) est vraie, cela signifie que R_{new} et R_{rev} ont au moins l'une des relations suivantes :

- Match parfait (R_{new} == R_{rev}) : le type de conflit est une observation complète (Algorithme 2: ligne 8).
- Sous-assemblage (R_{new} est un sous-ensemble de R_{rev}) : le type de conflit est l'observation partielle (Algorithme 2 : ligne 10).
- Généralisation (R_{rev} est un sous-ensemble de R_{new}) : le type de conflit est une exception (Algorithme 2 : ligne 12).
- Corrélation : Si aucune de ces relations précitées n'est vérifiée, le type de conflit est donc une corrélation.

5.5.3. Phase 3 : Résoudre le conflit

L'algorithme AGCSP-IPsec résout automatiquement les conflits Intra-Politique, sans aucune intervention manuelle. A cet effet, la technique PHPR est proposée. La principale caractéristique de cette solution est l'exemption de l'intervention de l'administrateur pour choisir la règle à omettre dans le cas où il est nécessaire de supprimer une règle de la SP.

5.5.3.1. Résolution des conflits Intra-Politique

Par souci de clarté, nous présentons un exemple simple d'une politique de sécurité notée P1, composée de 12 règles avec des conflits potentiels. Sur cet exemple, nous avons ajouté un nouveau prédicat en plus des clauses condition et action. Ce prédicat noté P, définit la priorité accordée à chaque règle de la politique (voir tableau 5.5).

Règle	Clause de Condition					Clause de l'Action		P
	Protocole	Src @ IP	Src Port	Dst @ IP	Dst Port	Action	Paramètres	
R1	TCP	192.168.1.12	*	173.13.0.0	25	Protect	Protect ESP Tunnel to GW_B	4
R2	Any	192.168.2.0/24	*	173.13.1.0/24	*	Deny	-	3
R3	TCP	192.168.1.4	*	10.0.0.1	25	Protect	Protect ESP Tunnel to GW_C	0
R4	Any	192.168.2.1	*	173.13.1.1	*	Deny	-	0
R5	TCP	192.168.1.12	*	173.13.0.0	25	Protect	Protect ESP Tunnel to GW_B	1
R6	TCP	192.168.0.0/16	*	173.13.0.0/24	25	Protect	Protect ESP Tunnel to GW_B	2
R7	TCP	192.168.1.4	*	10.0.0.1	25	Bypass	-	4
R8	TCP	192.168.1.0/24	*	10.0.0.0/24	25	Protect	Protect ESP Tunnel to GW_C	0
R9	TCP	192.168.2.3	*	173.13.1.1	25	Bypass	-	2
R10	TCP	192.168.2.2	*	10.0.0.0/24	25	Deny	-	1
R11	TCP	192.168.2.0	*	173.13.0.0	25	Deny	-	0
R12	TCP	192.168.2.0/24	*	10.0.0.1	25	Protect	Protect ESP Tunnel to GW_C	0

Tableau 5. 5 Exemple de politique de sécurité avec les priorités des règles

La résolution des conflits se fait selon leurs types comme suit :

- Ombrage complet : selon P1, l'ajout de la règle R7 provoque un conflit de type ombrage complet avec la règle R3. La résolution de ce conflit est simple ; la règle avec la priorité la plus basse est supprimée de la SP. Par conséquent, l'algorithme AGCSP défausse la règle R3 et ajoute la règle R7.
- Ombrage partiel : la règle R9 provoque un conflit d'ombrage partiel avec la règle R2. Pour résoudre ce conflit, la technique PHPR est utilisée. Si la nouvelle règle a une priorité plus élevée, elle prendra la préséance et sera déplacée vers le haut. Dans ce cas la nouvelle règle est considérée comme une exception intentionnée, sa priorité sera fixée à cinq (P = 5). Néanmoins,

la nouvelle règle est rejetée si elle a une priorité inférieure. Dans l'exemple de P1, la règle R9 est supprimée de la SP, car elle a une priorité inférieure à celle de R2.

- Exception : avant la résolution de ce conflit, l'algorithme AGCSP-IPSec vérifie s'il s'agit d'une exception intentionnée ou non. Le conflit sera ignoré si la priorité de la règle est égale à cinq (P = 5). Dans l'exemple de P1, la règle R7 est une exception de la règle R8. Par conséquent, AGCSP-IPsec supprimera la règle R7, car sa priorité est inférieure à cinq.
- Corrélation : dans l'exemple de P1, la règle R10 et la règle R12 sont corrélées. Pour résoudre ce conflit, une nouvelle règle est ajoutée à la politique. La nouvelle règle regroupe les prédicats communs entre les deux règles corrélées. En ce qui concerne la partie action, la nouvelle règle exécutera l'action associée à la règle ayant la priorité la plus élevée. La priorité de cette nouvelle règle est fixée à cinq (P = 5). Cette règle est alors considérée comme une exception intentionnée des deux règles corrélées R10 et R12.

Le résultat d'exécution de l'algorithme AGCSP-IPsec sur la politique P1 est donné sous la forme d'une nouvelle politique sans conflit présentée dans le tableau 5.6.

Rule	Clause de la Condition					Clause de l'action clause		P
	Protocol	Src IP	Src Port	Dst IP	Dst Port	Action	Setting	
R1	TCP	192.168.1.12	*	173.13.0.0	25	Protect	Protect ESP Tunnel to GW_B	4
R2	Any	192.168.2.0/24	*	173.13.1.0/24	*	Deny	-	2
R3	TCP	192.168.1.4	*	10.0.0.1	25	Bypass	-	5
R4	TCP	192.168.1.0/24	*	10.0.0.0/24	25	Protect	Protect ESP Tunnel to GW_C	0
R5	TCP	192.168.2.0/24	*	173.13.0.0/24	25	Bypass	-	2
R6	TCP	192.168.2.2	*	10.0.0.1	25	Deny	-	5
R7	TCP	192.168.2.2	*	10.0.0.0/24	25	Deny	-	1
R8	TCP	192.168.2.0/24	*	10.0.0.1	25	Protect	Protect ESP Tunnel to GW_C	0

Tableau 5. 6 Politique sans conflit générée par AGCSP-IPsec

5.5.3.2. Résolution des conflits Inter-Politique

Après la résolution de tous les conflits Intra-Politique, le processus de résolution des conflits Inter-Politique est déclenché. A chaque fois qu'un conflit est détecté, l'administrateur est averti et potentiellement sollicité pour intervenir à la résolution de conflit. Pour cette raison, les règles de la politique sont comparées aux politiques d'autres périphériques du même chemin. Un conflit Inter-Politique, peut être un conflit de contradiction (Algorithme 5.3 : ligne 12), non-pertinence (Algorithme 5.3 : ligne 14) ou bien un conflit d'invalidité (Algorithme 5.3 : ligne 16). Pour la résolution de ces conflits, l'administrateur choisit de remplacer les actions des règles ou de mettre à jour leurs conditions.

Algorithm 5.3. Inter-Policy conflict detection function

```
1:Begin
2:   Policy: P',P2; Rule: R;
3:   R= Get_Rule(P');
4:   while (R) do
5:        R= Get_Rule(P2);
6:        while (R) do
7:          if ("R.cond "∧ "R'.cond" ) then
8:             if ((R.action = "Deny") && (R'.action!= "Deny")) then
9:                Contradictory_resolution(P2,P',R,R');
10:           else if ((R.action = "Protect") && (R'.action!= Protect")) then
11:                 Irrelevance_resolution(P2,P',R,R');
12:                 else if  ((R.action = "Bypass") && (R'.action == "Deny"))
13:                   then Spuriousness_resolution(P2,P',R,R');
14:                    else Alert ("Rule Dropped");
15:               end if
16:           end if
17:         end if
17:       end while
18:   end while
19:end
```

Algorithme 5. 3 Pseudocode de la fonction de détection de conflit Intra-Politique

5.6. Evaluation des performances

Pour évaluer notre algorithme AGCSP-IPsec, nous avons effectué plusieurs expérimentations sur différentes politiques de sécurité. Ces expérimentations visent à garantir l'efficacité de notre algorithme et répondre aux issues relatives à la gestion des conflits des SP IPsec. Dans cette section, nous présentons d'abord l'environnement d'expérimentation utilisé. Ensuite, nous présentons les résultats

d'exécution de l'algorithme pour chaque expérimentation et on termine par une discussion des résultats obtenus.

5.6.1. Environnement d'évaluation

Afin d'évaluer AGCSP-IPsec, une implémentation de l'algorithme en C ++ est développée sur une machine Linux. La bibliothèque Buddy pour C / C ++ est utilisée pour permettre l'utilisation des opérations booléennes et manipuler les fonctions du diagramme de décision binaire ordonné (OBDD). L'implémentation est exécutée sur une machine avec un processeur Intel Core 2 Duos, avec 2 Go de RAM. Les expérimentations sont faites sur des politiques de sécurité qui varient de 50 à 500 règles, utilisée pour contrôler le trafic IPsec entrant et sortant. Nous visons dans cette section d'évaluer l'efficacité de notre algorithme ainsi que pour répondre à certaines questions pertinentes dans le contexte de gestion des SPs IPsec.

5.6.2. Résultats d'évaluation

L'évaluation de l'algorithme AGCSP-IPsec se compose de quatre expériences comme suit.

5.6.2.1. Première expérience

Nous visons à déterminer si les administrateurs réseau sont capables de détecter efficacement tous les conflits existants dans une politique de sécurité IPsec. Pour répondre à cette question, l'algorithme AGCSP-IPsec est appliqué sur plusieurs politiques de sécurité IPsec variant de 50 règles à 500 règles. Nous avons fait une comparaison entre le taux d'identification des conflits de l'algorithme AGCSP-IPsec et celui de l'administrateur réseau. Les résultats de cette expérience sont présentés sur la figure 5.4.

L'interprétation des résultats de cette expérience prouve que l'algorithme AGCSP-IPsec est plus efficace pour la détection des conflits. L'algorithme AGCSP-IPsec a pu détecter 131 parmi 140 conflits existants, ce qui représente un taux de détection de 93,6 %. Cependant, un administrateur réseau doté d'une expérience moyenne n'a pu détecter que 64 conflits (un taux de 48,9 %). De plus, on remarque que le nombre de conflits dans une politique de sécurité augmente proportionnellement au nombre de règles. Par conséquent, les politiques utilisées pour la gestion du trafic dans des réseaux IPsec de grande taille sont plus susceptibles aux conflits. Donc, on assume

que la vérification manuelle n'est pas efficace pour détecter tous les conflits existants dans une politique de sécurité IPsec. Cependant, l'algorithme AGCSP-IPsec est plus efficace pour cette tâche.

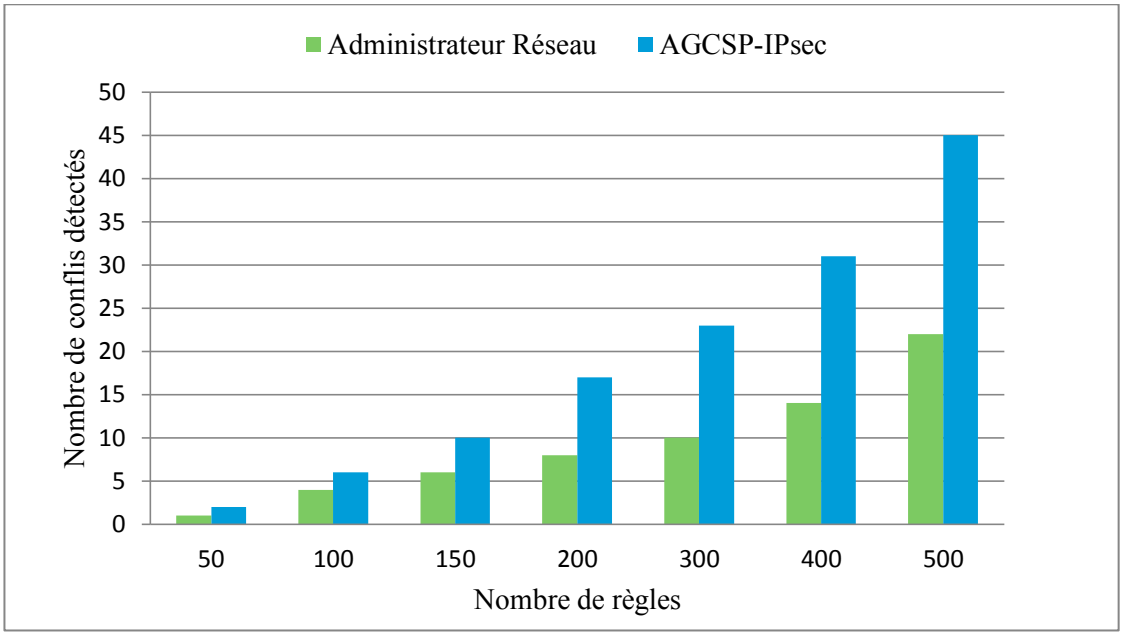

Figure 5. 4 Comparaison entre le taux de détection de conflits manuel et AGCSP-IPsec

5.6.2.2. Deuxième expérience

Nous avons effectué cette expérience dans le but d'identifier les conflits les plus courants dans une politique de sécurité IPsec. Pour cela, nous avons exécuté l'algorithme AGCSP-IPsec sur différentes politiques. La figure 5.5 présente le nombre d'apparitions de chaque type de conflits par rapport au nombre de règle de la politique en question. Nous avons calculé le taux moyen d'apparition de chaque type de conflit dans toutes les politiques vérifiées durant cette expérience (voir figure 5.6). L'analyse des résultats démontre que la redondance (régulière et inversée) et la duplication sont les conflits les plus courants dans la politique de sécurité IPsec. Ces conflits représentent 42 % de tous les conflits détectés. Les autres conflits sont moins fréquents, le conflit d'exception présente le type le moins courant avec un taux de 6%. On remarque aussi que les conflits Intra-Politique sont plus fréquents que les conflits Inter-Politique. Donc, ceci prouve que l'incohérence entre les règles de la même politique est plus fréquente. Il convient de mentionner que le nombre de conflits est relatif à la taille de la politique, c'est-à-dire au nombre de règles dans la politique de sécurité.

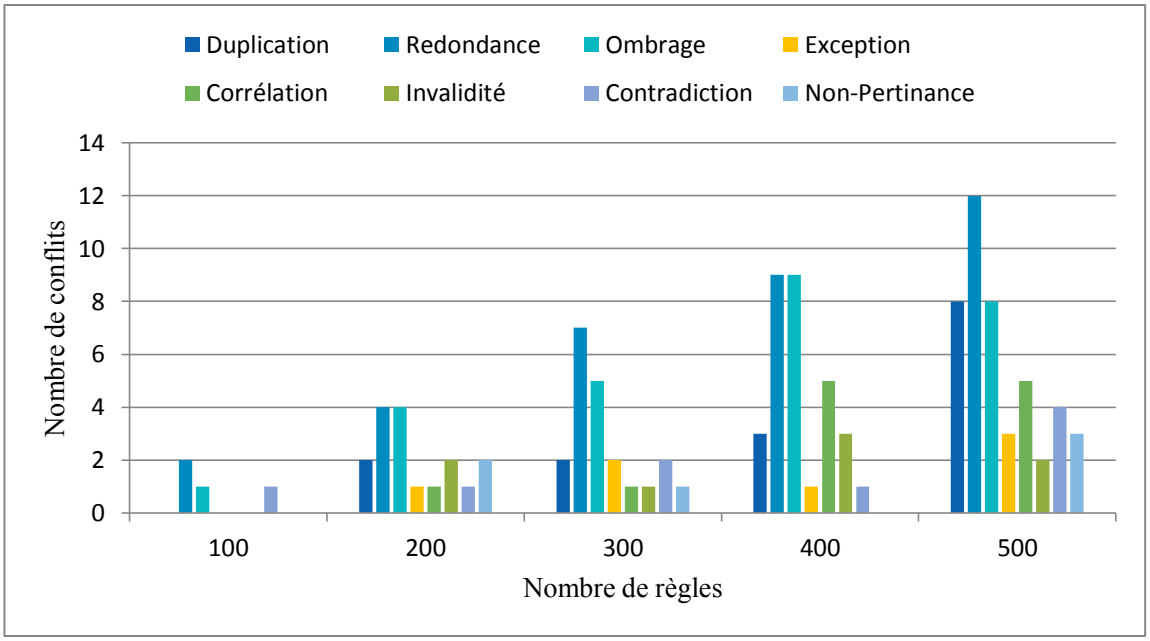

Figure 5. 5 Taux d'apparition de chaque type de conflit par rapport à la taille de la politique de sécurité

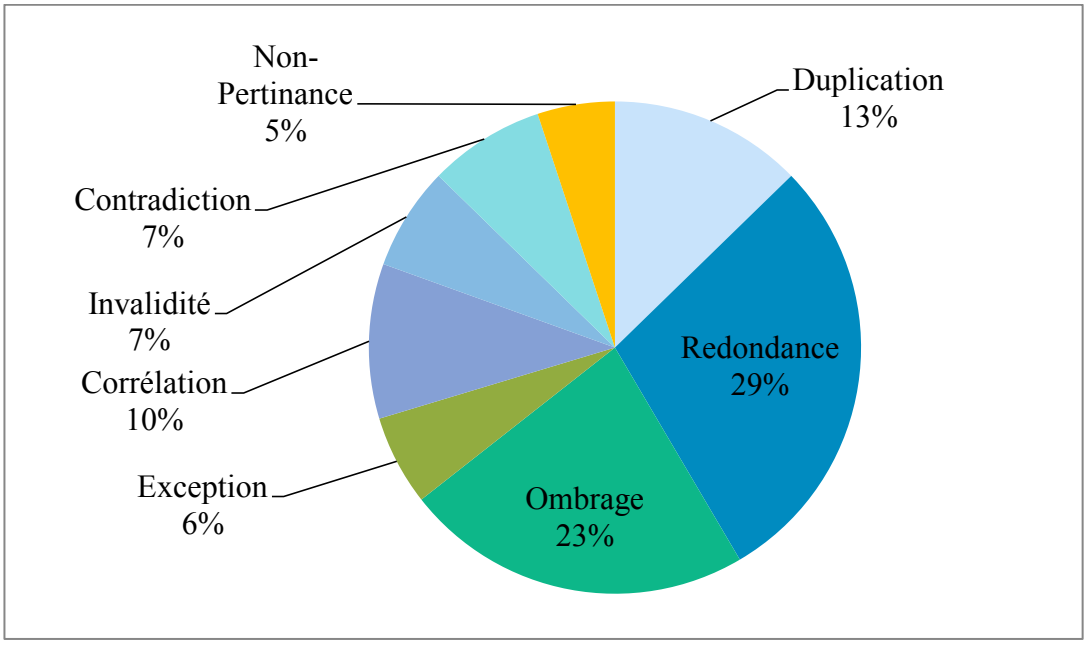

Figure 5. 6 Taux moyen d'apparition des conflits dans une politique de sécurité

5.6.2.3. Troisième expérience

La troisième expérience vise à démontrer l'importance de la vérification dynamique de la politique de sécurité. La vérification dynamique est assurée par notre méthode proposée basée sur la fonction F_{Total}. À cet effet, l'algorithme AGCSP-IPsec est mis à jour en éliminant F_{Total}. Par conséquent, la détection des conflits se fait d'une

manière statique ; l'algorithme AGCSP-IPsec applique une recherche itérative chaque fois qu'une nouvelle règle est ajoutée à la politique. Nous avons effectué une comparaison du temps de traitement nécessaire pour la vérification des différentes SP IPsec (composées de 50 à 500 règles), dans un premier lieu, en utilisant l'algorithme AGCSP-IPsec, et en deuxième lieu en utilisant la version modifiée (statique). Les résultats de cette expérience sont illustrés dans la figure 5.7.

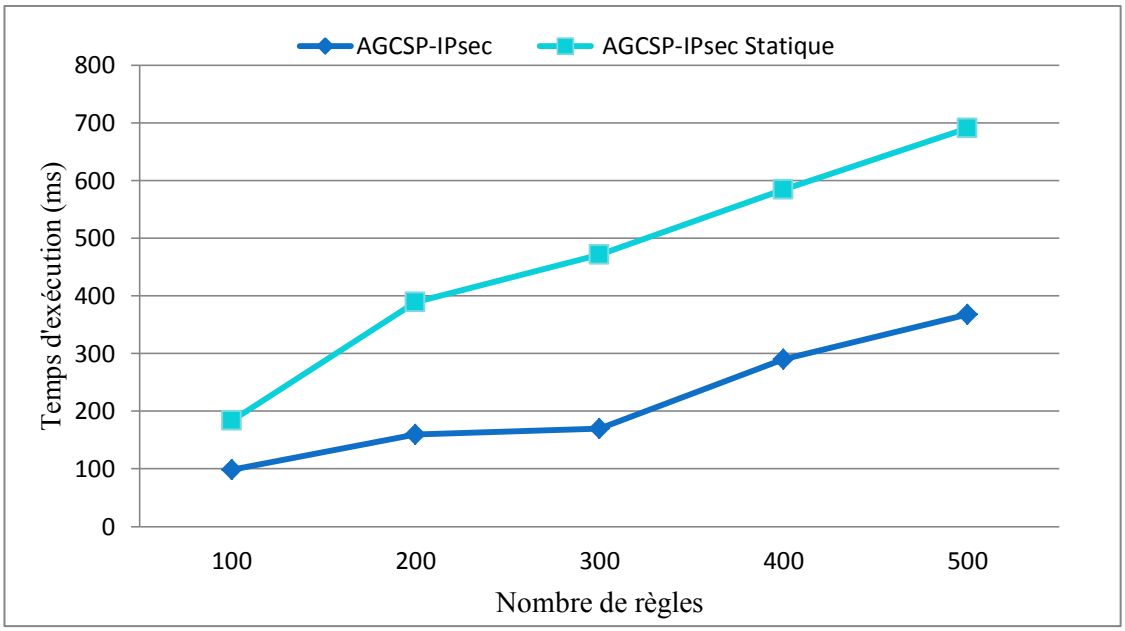

Figure 5. 7 Temps nécessaire pour la vérification d'une SP : AGCSP-IPsec VS AGCSP-IPsec statique

Les résultats montrent que l'algorithme AGCSP-IPsec, en version intacte, fonctionne plus rapidement que l'algorithme modifié. Ceci est dû à la méthode itérative qui prend beaucoup plus de temps pour la vérification de la politique. Sinon, la méthode dynamique assurée par AGCSP-IPsec est plus efficace et plus rapide, d'où on remarque qu'en moyenne cette méthode est deux fois plus rapide (un temps d'exécution moyen pour AGCSP-IPsec de 217 ms par rapport à 464 ms pour la méthode statique).

5.6.2.4. Quatrième expérience

Dans cette expérience, nous comparons les performances de l'algorithme AGCSP-IPsec avec l'un des travaux relatifs présenté dans le chapitre II. La comparaison est faite avec le travail de Niksefat et al. qui ont proposé l'algorithme IPCDR [51]. Les deux algorithmes ont été exécutés sur les mêmes politiques IPsec, allant de 50 à 500

règles. Les tests de performance de cette expérience sont: le temps de traitement et le taux d'utilisation de la mémoire. Une implémentation de l'algorithme IPCDR est utilisée pour cette comparaison. Étant donné que l'algorithme IPCDR est limité aux conflits Intra-Politique, les tests ont été exécutés uniquement sur SPs contenant des conflits Intra-Politique. Les figures 5.8 et 5.9 illustrent respectivement les résultats de la comparaison du temps de traitement et d'utilisation de la mémoire. En ce qui concerne l'espace mémoire utilisé dans la vérification des politiques, nous pouvons observer également que la consommation de mémoire croît proportionnellement avec la taille de la politique (nombre de règles). Cependant, qu'elle que soit la taille de la politique, AGCSP-IPsec consomme moins de mémoire par rapport à son concurrent.

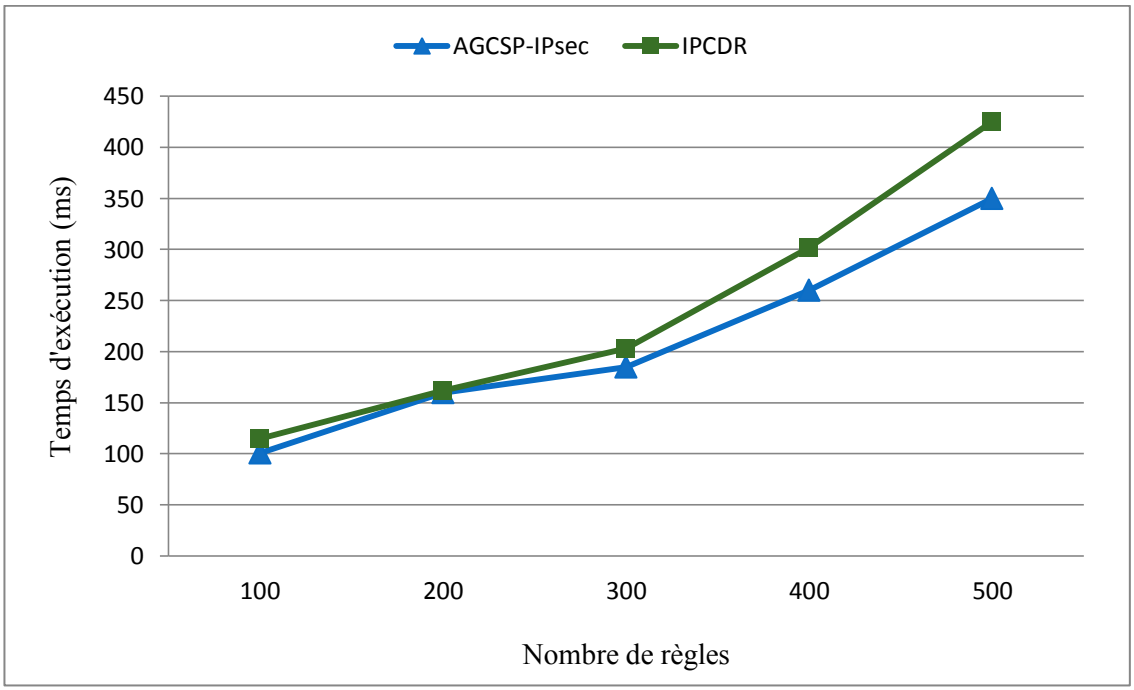

Figure 5. 8 Temps nécessaire pour la vérification d'une SP : AGCSP-IPsec Vs IPCDR [51]

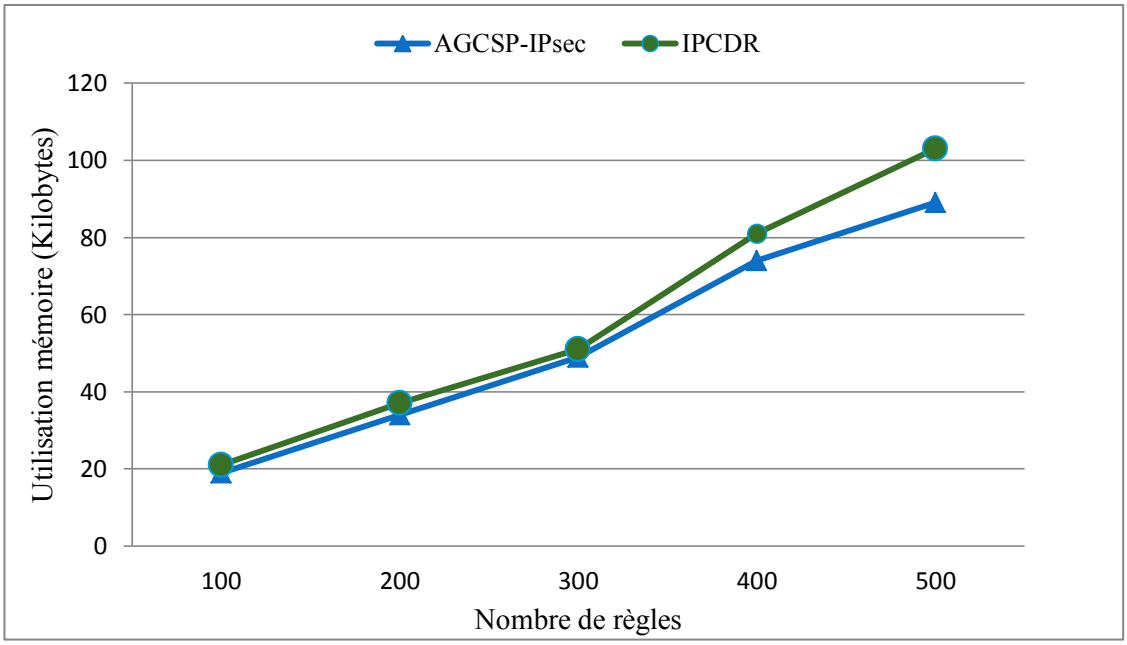

Figure 5. 9 Utilisation d'espace mémoire : AGCSP-IPsec Vs IPCDR [51]

IL est à noter que, cette proposition fut le sujet d'un article scientifique [119], qui a été soumis à un journal et il est en cours d'évaluation.

5.6.3. Comparaison avec autres travaux relatifs

Dans cette sous-section, nous comparons notre algorithme AGCSP-IPsec proposé avec d'autres travaux examinés dans le chapitre III, section 3.5.2. Le tableau 5.7 présente une comparaison entre ces travaux. Les critères de comparaison choisis sont : le type de la politique de sécurité, IPsec ou bien FW (firewall), la classification des conflits, la proposition d'mécanisme de détection et de résolution, la prise en compte des politiques dynamique. En terme de performance, la minimisation de temps d'exécution de la vérification ainsi que l'utilisation d'espace mémoire. La signification de notations utilisées dans le tableau est comme suit :

- NC : les auteurs ont proposé une nouvelle classification des conflits.
- UC : les auteurs ont mis à jour une classification existante ou ajouté une nouvelle catégorie de conflit.
- CE : la méthode des auteurs est basée sur la classification existante.
- ☑ Signifie : considéré.
- ☒ Signifie : non-considéré.

REF	Type de la SP	Classification des conflits	Détection des conflits	Résolution des conflits	Politique dynamique	Minimisation du temps d'exécution	Minimisation d'usage mémoire
[45]	IPsec	NC	☒	☒	☒	☒	☒
[46]	IPsec	NC	☑	☒	☒	☒	☒
[47]	IPsec	NC	☑	☒	☒	☒	☒
[48]	IPsec	NC	☑	☒	☒	☒	☒
[49]	IPsec	EC	☑	☒	☒	☒	☒
[50]	IPsec	NC	☑	☑	☒	☒	☒
[51]	IPsec	EC	☑	☒	☑	☑	☑
[56]	FW	Firewall	☑	☑	☑	☒	☒
[108]	FW	Firewall	☑	☑	☒	☑	☒
AGCSP -IPsec	IPsec	EC	☑	☑	☑	☑	☑

Tableau 5. 7. Comparaison entre les solutions proposées pour la gestion des politiques de sécurité IPsec

L'analyse des solutions présentées dans la sous-section précédente prouve l'efficacité et l'applicabilité de l'algorithme AGCSP-IPsec. Notre algorithme proposé répond aux besoins décrits au début de ce chapitre. Ainsi, les résultats des expériences menées ont approuvé l'importance de la vérification dynamique des politiques de sécurité. D'où, il a été prouvé que la gestion manuelle des administrateurs est inefficace quels que soient leurs niveaux d'expertise. Comme supposée, la vérification des SPs IPsec au cours des différentes expériences montre que les conflits sont plus fréquents dans les politiques de grande taille (grand nombre de règle). Ceci dit, le nombre de conflits augmente proportionnellement au nombre d'entités de réseau, ce qui rend la gestion automatique des politiques de sécurité un besoin crucial. Concernant les performances, l'algorithme AGCSP-IPsec a montré son efficacité. À savoir, la quatrième expérience évalue le temps de traitement et l'espace mémoire nécessaires pour une vérification complète d'une SP. Les tests révèlent que l'algorithme AGCSP-IPsec fonctionne mieux que son homologue.

5.7. Conclusion

Dans ce chapitre, nous avons présenté notre première contribution sur le protocole IPsec. Afin de résoudre le problème de la gestion des politiques de sécurité IPsec, nous avons proposé un algorithme efficace de gestion des politiques (AGCSP-IPSec). AGCSP-IPsec peut non seulement satisfaire aux exigences des politiques de sécurité, mais peut aussi détecter et résoudre les conflits éventuels. Nous avons démontré qu'AGCSP-IPsec est efficace en termes de temps de traitement et d'usage mémoire. D'où, l'algorithme proposé est adapté aux environnements dynamiques (les réseaux distribués, IoT, WSNs) où les performances sont très importantes. Par conséquent, AGCSP-IPsec est un moyen efficace qui peut être utilisé facilement pour remplacer la gestion manuelle des SPs IPsec dans de tels environnements.

CHAPITRE VI

TAKE-IoT : Protocole d'échange de clé authentifié pour l'IoT

Dans le chapitre précédant, nous avons présenté l'algorithme AGCSP-IPsec pour la gestion automatique des politiques de sécurité IPsec dans des environnements dynamiques. Dans ce chapitre, nous nous sommes intéressés à un autre composant de l'architecture de sécurité d'IPsec, qui est les associations de sécurité. Les associations de sécurité définissent les paramètres de sécurité nécessaires pour sécuriser une communication IPsec. Les pairs communicants doivent négocier ces paramètres avant de se mettre d'accord sur une association de sécurité. Ces négociations se font à travers des échanges sécurisés, dont leur sécurité dépend essentiellement du protocole IKEv2, le protocole standard d'échange des clés pour IPsec. Cependant, l'utilisation d'IPsec dans des environnements dynamiques, comme l'Internet des objets (IoT), a besoin de quelques adaptations (allégement et compression) pour convenir à des environnements tels que les réseaux 6LoWPANs qui présentent un élément essentiel de l'IoT. Ces adaptations s'appliquent également sur l'ensemble des protocoles sous-jacents d'IPsec, comme le protocole IKEv2. Dans cette deuxième contribution, nous proposons un protocole léger d'échange de clé authentifié pour l'IoT, en anglais : Tiny Authenticated Key Exchange Protocol for IoT (TAKE-IoT). Le protocole proposé vise à remplacer IKEv2. TAKE-IoT fonctionne avec IPsec pour assurer une gestion dynamique et sécurisée des clés et d'association de sécurité entre les nœuds capteurs et les hôtes IPv6.

Ce chapitre est organisé comme suit : la section 6.1 introduit le chapitre en présentant l'environnement visé et les objectifs de la recherche que notre proposition doit accomplir. La section 6.2 introduit des préliminaires sur les fonctions de hachage et la méthode de signature utilisée dans notre proposition. Nous présentons notre protocole TAKE-IoT dans la section 6.3. La section 6.4 est dédiée à analyse de sécurité du protocole proposé. Nous comparant également une le protocole avec les mécanismes relatif existant dans la littérature. Finalement, la section 6.5 conclut ce chapitre.

6.1. Introduction

L'Internet des objets (IoT) définit la capacité d'intégrer des objets hétérogènes du monde réel à Internet. Ce paradigme vise à exploiter des objets intelligents pour effectuer des tâches quotidiennes de l'humain. Ces objets sont généralement caractérisés par une faible

puissance (alimentés par batterie) et par des faibles capacités de calcul. Dans le contexte de l'IoT, les réseaux de capteurs sans fil (WSN) prolifèrent dans la vie quotidienne sous forme de différentes applications, telles que la Cyber-santé [109], la domotique [110] et le contrôle du trafic [111]. Dans l'Internet d'aujourd'hui, les objets sont principalement des serveurs, des commutateurs, des pare-feu et des routeurs, des ordinateurs portables, des téléphones et des tablettes ... etc. Par conséquent, ces objets ont besoin d'une adresse IP pour la connectivité IP. Les réseaux de capteurs sans fil connectés à IP (WSN IP-connecté) sont considérés comme la nouvelle technologie sous-jacente pour l'IoT. Ils peuvent être étroitement intégrés aux infrastructures IP existantes à l'aide de la technologie 6LoWPAN. Cependant, l'issue de sécurité dans de tels réseaux est encore un sujet controversé.

Sur la base de l'analyse de la littérature existante présentée dans le chapitre III, nous avons démontré qu'IPsec est faisable pour les WSNs IP-connecté grâce aux opérations de compression. Néanmoins, l'échange de clés reste une question équivoque. Par défaut, IPsec gère les clés de sécurité à l'aide du protocole IKEv2. Cependant, ce protocole lourd ne convient pas aux appareils IoT à ressources limitées. Pour ces raisons, l'objectif principal de cette contribution est de développer un protocole d'échange de clé sécurisé et à faible coût pour IPsec. Nous envisageons de remplacer IKEv2 par un protocole d'échange de clés amélioré, conçu pour répondre aux contraintes des réseaux 6LoWPANs. Par conséquent, les objectifs de cette proposition sont :

- Le protocole TAKE-IoT proposé doit fonctionner avec IPsec pour assurer une gestion dynamique et sécurisée des clés et des associations de sécurité entre les nœuds capteurs et les hôtes IPv6.
- TAKE-IoT doit être un protocole léger et n'utilise que des opérations simples pour la génération des clés, afin de s'adapter aux contraintes des dispositifs IoT.
- TAKE-IoT doit remplir les propriétés de sécurité requises telles que : confidentialité persistante, le contrôle de clé et la sécurité de clé connue… etc.
- Le protocole proposé doit fournir les fonctionnalités de sécurité souhaitées et résister aux attaques qui menacent les réseaux de capteur, telles que les attaques

par réflexion, les attaques DOS, les attaques de relecture et les attaques de l'homme au milieu (MITM).

En plus de ces objectifs, le protocole proposé doit assurer les propriétés de sécurité requise pour l'IoT qui sont :

- La confidentialité : assurer que les données échangées ne sont pas accessibles, lues ou utilisées par des utilisateurs non autorisés (applications, processus, systèmes ou humains).
- L'authentification et l'intégrité : l'authentification assure l'identité des entités du réseau. Quant à l'intégrité, c'est la capacité assurée qu'un système et ses données n'ont pas subi de modifications non autorisées.
- Distribution de clé : permet l'échange sécurisé et efficace des clés.
- Perfect Forward Secrecy (PFS) (confidentialité persistante): s'assurer que si la clé privée est compromise, il n'y aura aucun moyen de compromettre les clés de session.
- Efficacité : assure que les mécanismes utilisés dans le system répondent aux objectifs et aux exigences annoncés.
- Sécurité des clés connues: assure que la clé utilisée pour chiffrer et déchiffrer les messages ne peut pas être divulgué par un tiers non-autorisé.
- Contrôle de clé : assure qu'aucune partie non-autorisée ne peut forcer la clé de session à une valeur présélectionnée.
- Fraîcheur des données : assure qu'il n'y a pas un moyen de rejouer les messages déjà envoyés.

6.2. Préliminaires

Dans cette section, nous présentons des préliminaires, y compris les fonctions de hachage et les signatures à courbe elliptiques.

6.2.1. Fonction de hachage

Les fonctions de hachage cryptographique sont utilisées dans les applications qui fournissent l'authentification et l'intégrité. Ces fonctions sont des algorithmes qui convertissent des messages de taille arbitraire en appliquant un ensemble de transformations pour qu'ils renvoient en sortie un texte de taille fixe, appelée condensé ou code de hachage. Il existe de nombreuses fonctions de hachage tels que l'algorithme de hachage sécurisé (SHA-1, 2 et 3), Message-digest (MD5), RIPEMD…, etc.

Parmi celles-ci, l'algorithme SHA-3 introduit l'utilisation de constructions en éponge (Sponge-Based Construction). La construction Merkle–Damgard ou la construction en éponge [112] est utilisée pour la construction des fonctions de hachage. Les fonctions éponge sont utilisées pour générer des codes de hachage de longueur arbitraire. Une fonction éponge entraîne une construction qui prend une entrée de longueur variable. Elle utilise une fonction de permutation « f » de longueur fixe et fonctionne sur un nombre de bit « b » donné. La largeur du bit « b » (b_bit width) est appelée comme un « état » dans la construction de l'éponge [113]:

$b = r + c$. Où « r » est le débit binaire et « c » est la capacité.

Le message d'entrée est divisé en blocs de « r » bits chacun, où un rembourrage est effectué si nécessaire. Il y a deux phases dans la construction de la fonction éponge, la phase d'absorption et la phase de compression. Dans la phase d'absorption, l'algorithme applique un XOR sur les premiers « r » bits de l'état et de l'entrée. Il traite ces bits avec la fonction de permutation « f ». Ce processus est répété pour tous les blocs d'entrée. Dans la phase de compression, les premiers « r » bits de l'état sont extraits du bloc de sortie. En fonction du nombre de bits en sortie, le nombre de blocs dans la phase de compression augmente. Seuls les premiers « r » bits sont directement affectés par l'entrée. Les « c » derniers bits de l'état restent inchangés. Cependant, ils ne restent pas comme valeur constante en raison de la fonction de permutation « f ». Donc, la valeur de « c » détermine le niveau de sécurité atteint par la construction.

La fonction de hachage utilisée dans IKEv2 est la fonction SHA. Cependant, dans notre contribution, nous proposons d'utiliser une fonction de hachage qui suit la construction

en éponge. Nous avons choisi la fonction MBLAKE [114] pour notre protocole. MBLAKE est une version modifiée de la fonction de hachage BLAKE l'une des finalistes du deuxième tour des compétitions SHA-3 menées par l'institut national des standards et technologie (NIST) [115]. MBLAKE fournit de meilleurs résultats sur le plan des calculs des codes de hachage ainsi que pour générer et vérifier les signatures de courbe elliptique [105]. Par conséquent, elle convient mieux aux environnements à ressources limitées tel que l'IoT.

6.2.2. Les signatures avec courbe elliptiques

Notre proposition est basée sur la cryptographie à courbes elliptiques (ECC). L'utilisation de l'ECC garantit la sécurité avec des clés de longueur courte, ce qui minimise l'espace de stockage des clés. Les courbes elliptiques sont également utilisées pour les signatures numériques comme dans l'algorithme ECDSA basé sur la cryptographie à clé publique.

Les clés publiques et privées sont générées en fonction des paramètres de domaine de la courbe spécifique. Les clés privées sont des entiers choisis aléatoirement de l'intervalle [1..n-1], où « n » est l'ordre premier du sous-groupe. Dans notre travail, une version modifiée d'ECDSA est utilisée comme présenté dans la section 6.3.2.

6.3. Notre protocole proposée TAKE-IoT

Cette section décrit le protocole TAKE-IoT proposé et met en évidence ses caractéristiques. Le protocole est conçu pour résoudre le problème de l'établissement de clés et d'échange d'association de sécurité pour les appareils IoT à ressources limitées, tels que les dispositifs 6LoWPANs ou ce qu'on note : les WSNs IP-connecté.

6.3.1. Caractéristique du protocole

TAKE-IoT vise à assurer une haute sécurité contre les attaques tout en réduisant les coûts de calcul et la consommation d'énergie. Le protocole proposé est conçu pour être utilisé par IPsec, comme un protocole d'échange de clés au lieu du protocole IKEv2. L'utilisation de TAKE-IoT avec IPsec au niveau de la couche réseau vise à fournir une

sécurité de bout en bout (E2E) efficace entre les nœuds de capteur et les hôtes IPv6. Le protocole convient à un environnement où les transmissions de paquets sont périodiques ou transmises en fonction de l'action de l'utilisateur. Les données seront transmises entre un routeur 6LoWPAN et un dispositif d'implémentation minimal, tel qu'un capteur de température ambiante. (voir figure 6.1).

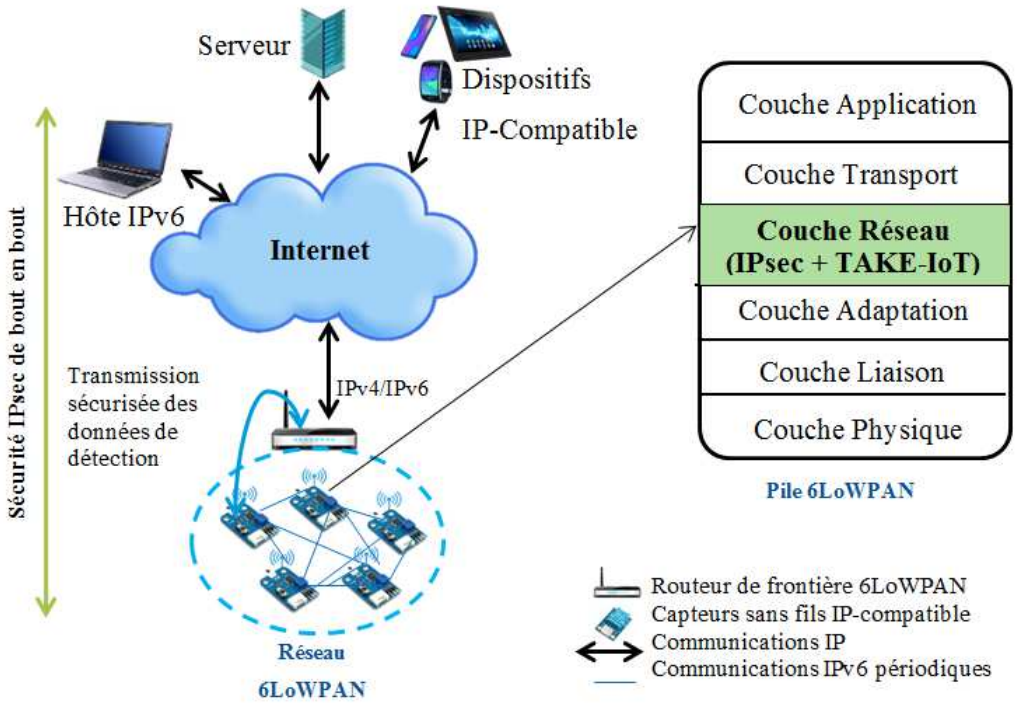

Figure 6. 1 Positionnement de TAKE-IoT par rapport à l'architecture du réseau

TAKE-IoT se caractérise par un échange d'une seule association de sécurité (TAKE_IoT_SA). Cet échange est accompli en une seule phase, composée de 3 messages échangés entre l'initiateur et le répondeur. TAKE-IoT transmet les paquets ESP à la destination connue, mais lorsque l'appareil passe en mode veille, il ne conserve pas l'association existante, puisque elle sera recréée lors de la prochaine transmission. L'initiateur répond avec un message vide (ou d'erreur) à toutes les autres demandes optionnelles entrantes. Les fonctionnalités optionnelles d'IKEv2 telles que NAT Transversal, IKE_SA re-keying, l'authentification EAP, les cookies…, etc. sont exclues.

En ce qui concerne le mécanisme d'authentification, TAKE-IoT utilise une authentification par secret partagé. Par conséquent, l'ID de fournisseur, le certificat, la demande de certificat et les charges utiles de configuration sont tous ignorés.

L'ECC est utilisée pour fournir des signatures d'authentification. Une version modifiée d'échange de clés Diffie–Hellman à courbe elliptique est utilisé pour l'échange de clé. Pour la fonction de hachage, TAKE-IoT utilise la fonction MBLACK présentée dans la section précédente.

Concernant le chiffrement et la génération des clés, le protocole est basé sur ECC. Les paramètres de domaine des courbes elliptiques sont convenus par les deux entités communicantes. Ces informations sont incluses dans l'association de sécurité proposée par l'initiateur. L'initiateur et le répondeur décident de leurs clés privées et génèrent les clés publiques correspondantes. Donc, TAKE-IoT est basé sur la cryptographie à clé publique.

6.3.2. Description du protocole

Afin de répondre aux exigences des environnements IoT, le protocole TAKE-IoT réduit les charges utiles, l'échange de messages et le nombre d'opérations effectuées. TAKE-IoT a une seule phase, qui ne comprend que trois messages, échangés entre l'initiateur (I) et le répondeur (R), notés M1, M2 et M3. Les notations et les opérations utilisées dans l'algorithme de TAKE-IoT sont présentées dans le tableau 6.1.

Notations	**Description**
ID_I	Identité de l'initiateur
ID_R	Identité du répondeur
$SA_{proposal}$	les algorithmes cryptographiques suggérés par l'initiateur
SA_{chosen}	Suite cryptographique choisie par le répondeur à partir des suggestions offertes par l'initiateur.
P	Le point générateur d'ECC
t_I, t_R	Clés privées statiques de l'initiateur et du répondeur
T_I, T_R	Clés publiques statiques de l'initiateur et du répondeur

v_I, v_R	clés privées de l'initiateur et du répondeur
V_I, V_R	clés publiques de l'initiateur et du répondeur
K	Clé de session éphémère calculée par l'initiateur et le répondant
K_{IR}	La clé de session dérivée par l'initiateur et le répondeur
E_{KIR} {M}	Cryptage du message M avec utilisation d'un système de cryptage symétrique par K_{IR}
H (M)	Hachage du message M avec de la fonction de hachage MBLAKE
||	Opération de concaténation
.	Produit scalaire

Tableau 6. 1 Description des notations et des charges utiles

Notre protocole est composé de trois messages, les deux premiers messages sont utilisés pour échanger les clés publiques ainsi qu'établir l'association de sécurité. C'est-à-dire, se mettre d'accord sur les algorithmes cryptographique et les paramètres de sécurité qui seront utilisées dans cette communication. Après, dans le troisième message, les deux paires s'authentifient et vérifient leurs identités. C'est la vérification est correcte, ils commencent l'échange des données secrètes. Ces données sont chiffrées par les clés secrètes générées durant la négociation. Les messages échangés sont détaillés ci-dessous.

M1: Initiateur → Répondeur: $T_I, ID_I, SA_{Proposal}$

Avant l'envoie du premier message l'initiateur effectue les opérations suivantes :

- Sélectionne l'association de sécurité proposée $SA_{Proposal}$.
- Sélectionne un nombre secret aléatoire comme clé privée $v_I \in [1, n-1]$ où n est un grand nombre premier.
- Calcule $V_I = v_I . P$,
- Sélectionne un deuxième nombre secret aléatoire comme sa clé privé statique $t_I \in [1, n-1]$.
- Calcule sa clé publique: $T_I = H(t_I \| v_I) . P$

Par la suite, l'initiateur envoie TI, IDI et les algorithmes cryptographiques proposés (SAproposal) au répondeur.

M2: Répondeur → Initiateur: T_R, E_{KIR} {ID_I || ID_R || V_R || L_R || SA_{Chosen} || H (SA_{Chosen})}

En recevant le message de la part de l'Initiateur, le Répondeur à son tour exécute les opérations suivantes :

- Sélectionne un nombre secret et aléatoire $v_R \in [1, n-1]$ et calcule $V_R = v_R . P$
- Sélectionne un deuxième nombre secret et aléatoire comme sa clé privée statique $t_R \in [1, n-1]$.
- Calcule sa clé publique :
  - $T_R = H (t_R \| v_R) . P$
  - Et calcule aussi :
  - $E = H (T_R)$
  - $K = H (t_R \| v_R) . T_I$
  - $L_R = V_R^{-1}(E + T_R . K) \bmod n$
  - $K_{IR} = H (ID_I \| ID_R \| X_{TI} \| X_{TR} \| X_K)$, *où X_{TI}, X_{TR} et X_K sont respectivement les x-coordonnées de T_I, T_R et K.*

En utilisant la clé générée K_{IR}, le Répondeur chiffre l'ensemble de ses algorithmes cryptographiques convenus pour la TAKE-IoT_SA (SA_{chosen}) et le digest H (SA_{chosen}), en plus de son identité et celle de l'initiateur, accompagné de V_R et L_R. Finalement, il envoie à l'initiateur T_R avec ces charges utiles chiffrées.

M3: Initiateur → Répondeur: E_{KIR} {V_I || L_I || $SA_{proposal}$ || H ($SA_{proposal}$)}

Lors de la réception du message du Répondeur, l'initiateur calcule :

- $K = H (t_I \| v_I) . T_R$
- $K_{IR} = H (ID_I \| ID_R \| X_{TI} \| X_{TR} \| X_K)$, *où X_{TI}, X_{TR} et X_K sont les x-coordonnées de T_I, T_R et K respectivement.*
- déchiffre les charges utiles chiffrées qu'il les a reçus, en utilisant la clé K_{IR}. Si la clé ne convient pas au déchiffrement, la session est terminée.
- Ensuite, il calcule:
  - $I = L_R^{-1} \bmod n$,
  - $E = H (T_R)$,

- $Z= (E. I) \bmod n$,
- $W = (K.I) \bmod n$,
- $X = Z. P + W. T_R. P$
- $V_R= E. L_R^{-1}.P+K.L_R^{-1}.T_R. P$

➢ L'initiateur vérifie si $V_R = X$, si c'est le cas:

- $E=H (T_I)$,
- et
- $L_I=v_I^{-1}(E+T_I. K) \bmod n$, Sinon la session est terminée.

L'initiateur utilise la clé K_{IR} pour chiffrer l'ensemble de ses algorithmes cryptographiques convenus pour TAKE-IoT_SA ($SA_{proposal}$) et le digest H ($SA_{proposal}$), accompagné de V_I et L_I.

Lors de la réception du message de l'initiateur, le répondeur vérifie si SAproposal est identique à celle envoyée dans M1. En plus, il vérifie que V_I calculé est le même que celui reçu. Une fois la vérification réussie, le répondeur assure l'identité de l'initiateur et commence l'échange des données, sinon la session se termine. Les étapes détaillées du TAKE-IoT sont présentées dans la Figure 6.2.

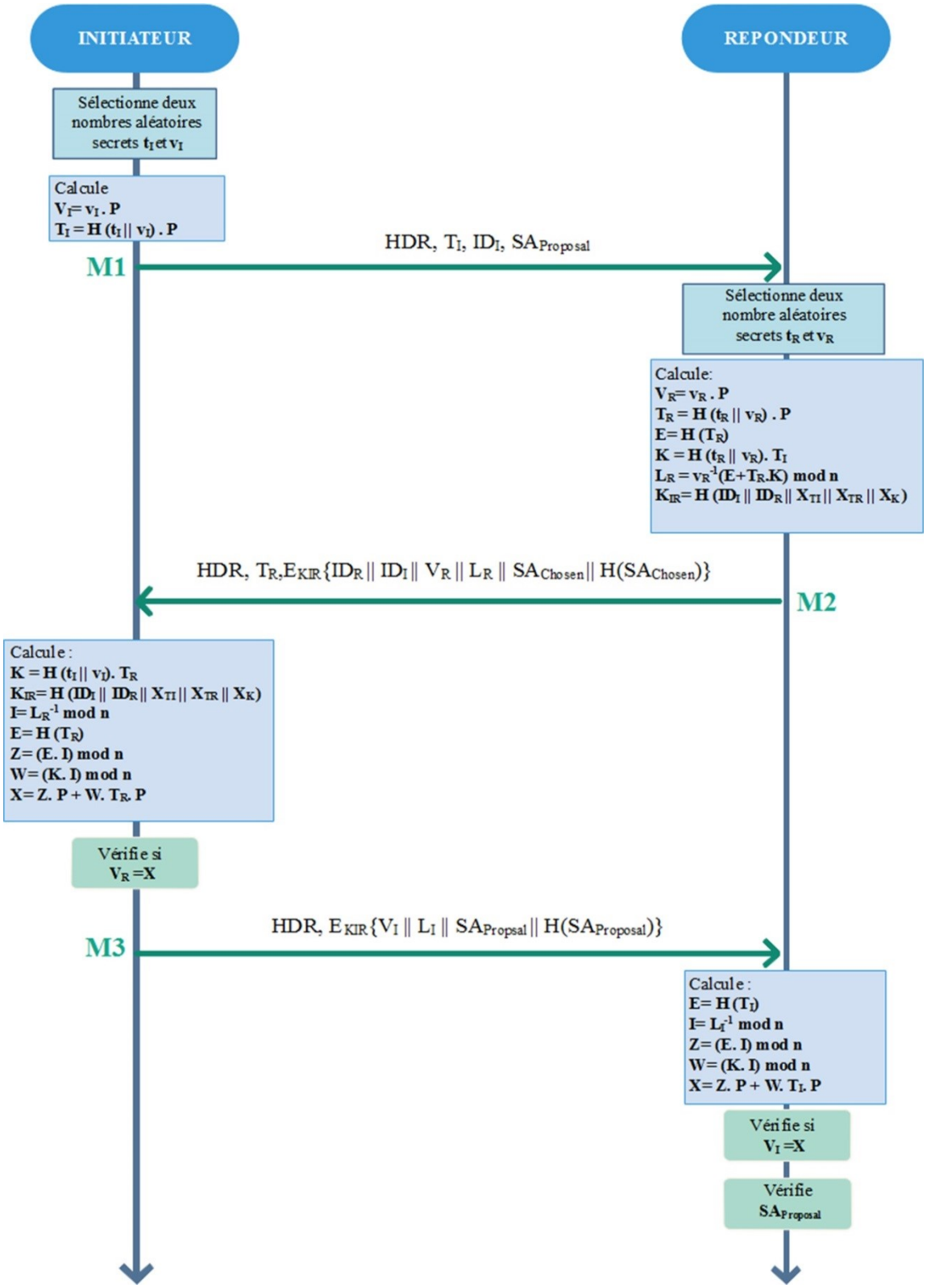

Figure 6. 2 L'échange de messages du protocole TAKE-IoT

6.4. Analyse de la sécurité

Dans cette section, nous analysons d'abord le protocole TAKE-IoT en fonction des objectifs de sécurité présentés dans la section 6.1. Ensuite, nous effectuons une vérification formelle en utilisant l'outil de validation automatisée des protocoles et applications de sécurité Internet (AVISPA). Cette vérification vise à confirmer la sécurité du schéma proposé et sa résistance aux attaques. En outre, nous présentons une comparaison avec les autres travaux relatifs et on termine par une discussion qui récapitule la section.

6.4.1. Analyse théorique

Notre solution est proposée pour l'IoT, et particulièrement les réseaux 6LoWPANs qui sont composés essentiellement des WSN IP-connecté. Par conséquent, les exigences de sécurité des réseaux de capteurs doivent être prises en compte.

6.4.1.1. Satisfaction des propriétés de sécurité

TAKE-IoT satisfait toutes les propriétés de sécurité sont essentielles dans les réseaux de capteurs :

- **Confidentialité des données :** les messages d'échange de clés sont cryptés à l'aide de clés partagées secrètes. Donc, ceci garantit la confidentialité des données transmises.
- **Authentification et intégrité des données :** TAKE-IoT assure une authentification mutuelle entre l'initiateur et le répondeur. Cela se fait lorsque l'initiateur (M2) et le répondeur (M3) vérifient la similitude entre le X calculé et les demandes d'authentification (V_I et V_R). En outre, l'utilisation des fonctions de hachage garantit l'intégrité des données.
- **Distribution de clés :** cette exigence est accordée par le protocole d'échange de clés proposé TAKE-IoT. En effet, la distribution des clés est l'objectif fondamental de TAKE-IoT.
- **Perfect Forward secret :** la satisfaction de cette propriété garantit que même si les clés à long terme sont compromises, cela ne doit pas conduire à la compromission des clés de session précédentes. Dans le protocole TAKE-IoT,

les clés privées: t_I, t_R, v_I et v_R utilisées pour les calculs de la clé K_{IR}, sont choisies au hasard. Par conséquent, la clé partagée K_{IR} ne peut pas être utilisée pour dériver d'autres clés de session. Donc, un adversaire ne peut pas récupérer K_{IR} et ne peut donc pas récupérer les messages précédemment envoyés. Ainsi, TAKE-IoT satisfait la propriété Perfect Forward Secrecy.

- **Efficacité :** TAKE-IoT utilise le système de courbe elliptique, qui offre une sécurité avec des clés de courte longueur (128 bits). Par conséquent, offrant moins d'espace de stockage et une charge de calcul inférieure. De plus, le protocole proposé ne nécessite que trois messages pour établir une association de sécurité. En outre, seules les opérations simples telles que l'addition et la division.
- **Sécurité des clés connues :** le protocole TAKE-IoT garantit cette propriété, car la clé est calculée à l'aide de nombres aléatoires, variables et difficiles à calculer. Par conséquent, une clé de session bien protégée, unique et aléatoire est créée pour chaque exécution.
- **Contrôle de clé :** TAKE-IoT ne permet à aucune partie de forcer la clé de session partagée à une valeur présélectionnée.
- **Fraîcheur des données :** garantie ; car chaque session a ses propres clés privées t_I, t_R, v_I et v_R et une clé partagée unique K_{IR}. Par conséquent, TAKE-IoT garantit qu'aucun adversaire n'a rejoué les anciens messages.

6.4.1.2. Résistance aux attaques

La dernière partie de cette analyse théorique met l'accent sur les attaques les plus courantes contre les réseaux de capteurs. Ainsi, nous démontrons comment TAKE-IoT peut résister à ces attaques.

- **Attaque par rejeu (Replay) :** TAKE-IoT est résistant à ce type d'attaque. Une attaque par rejeu se produit lorsqu'un adversaire espionne une conversation entre l'initiateur et le répondeur. Cet adversaire communique avec le répondeur et lui fait lire TI de la dernière session. Par conséquent, le répondeur lui renvoie TR et des charges utiles chiffrées. Cependant, en arrivant au troisième message M3,

l'adversaire ne pourra pas calculer la clé secrète partagée KIR et il ne pourra pas produire le LI qui correspond à TR. Par conséquent, l'attaque ne peut réussir.

- **Attaque d'homme au milieu (MITM) :** Le protocole proposé résiste à cette attaque en raison de l'authentification mutuelle entre l'initiateur et le répondeur. Ceci est garanti lors de la réception du M2, où l'initiateur vérifie si X = VR et lors de la réception du M3, où le répondeur vérifie à son tour X = VI. Donc, même si l'adversaire "A" peut remplacer la demande d'authentification VR par VA, cette attaque ne réussira pas.
- **Attaque par déni de service (DOS) :** TAKE-IoT a deux types de messages d'inondation : M1 et M3. En ce qui concerne M1, un attaquant choisit au hasard les composants du premier message, et forge un message falsifié afin d'inonder le répondeur et de gaspiller les ressources. Cependant, les opérations effectuées par le répondeur sont assez simples et pourraient être effectuées rapidement. Par conséquent, une attaque par déni de service ne peut pas empêcher le service offert par le répondeur, sauf dans le cas où l'attaque s'exécute pendant une période suffisamment longue. Concernant M3, l'initiateur envoie au répondeur les paramètres d'association de sécurité proposés ainsi que LI et VI chiffrés avec la clé de session KIR. Une attaque DoS peut être effectuée si un attaquant intercepte M3 et modifie les paramètres de sécurité afin de déclasser les services du répondeur. TAKE-IoT empêche les attaques DoS en utilisant M3 grâce à la clé partagée KIR utilisée pour crypter les charges utiles du M3. Donc, un attaquant ne peut pas forger ce message pour perturber le service.
- **Attaque par réflexion :** TAKE-IoT empêche cette attaque grâce à son mécanisme d'authentification. Par conséquent, l'initiateur vérifie l'identité du répondant dans M2 et le répondant vérifie à son tour l'identité de l'initiateur dans le troisième message M3.

6.4.2. Vérification Formelle

Afin de valider la sécurité du protocole proposé, cette sous-section présente une vérification formelle à l'aide de l'outil de validation automatisée des protocoles et applications de sécurité Internet (AVISPA) [116].

6.4.2.1. Présentation d'outil de vérification

En fait, AVISPA est l'un des outils de validation de sécurité automatisés les plus couramment utilisés. Il fournit une interface graphique aux utilisateurs comme le montre la figure 4, le format de sortie (OF) d'AVISPA est généré à l'aide de l'un des quatre back-end suivants (Voir Figure 6.3): On-the-fly Model-Checker (OFMC), Constraint Logic based Attack Searcher (CL-AtSe), SAT-based Model-Checker (SATMC) et Tree Automate based on Automatic Approximations for the Analysis of Security Protocols (TA4SP). A noter qu'actuellement, seuls les back-end OFMC et CL-AtSe peuvent gérer les propriétés algébriques et l'exponentiation Diffie-Hellman.

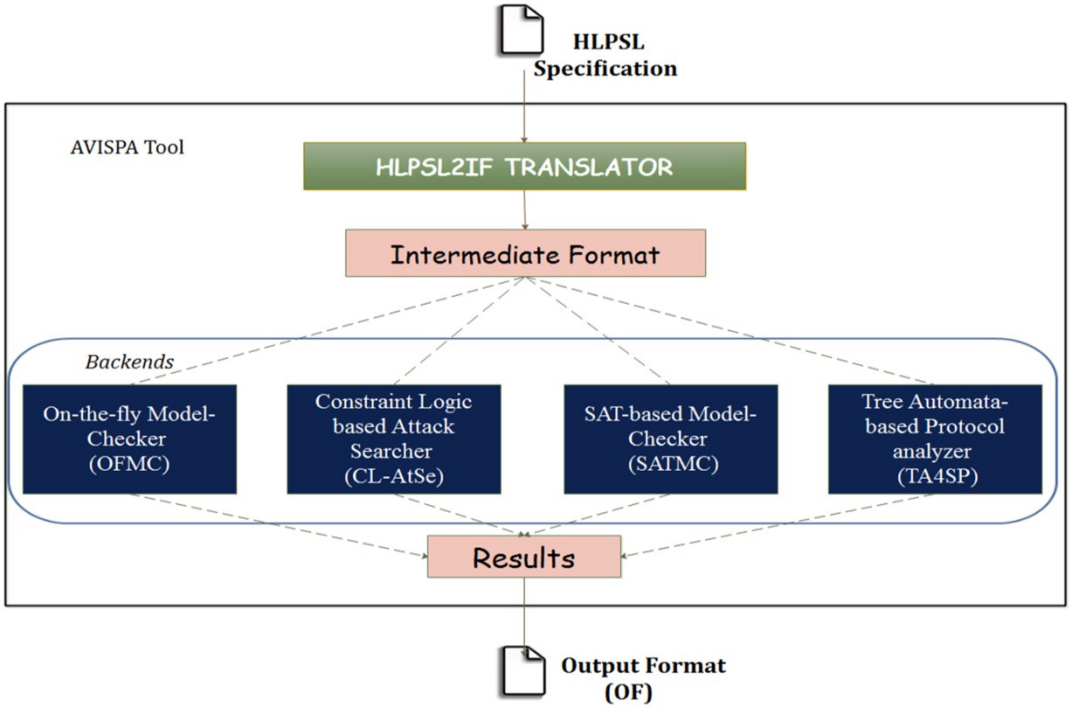

Figure 6. 3 Architecture de l'AVISPA

AVISPA utilise un langage de spécification de protocole de haut niveau (HLPSL) pour modéliser un protocole de sécurité. Une fois qu'un protocole est introduit dans AVISPA et modélisé en HLPSL, il est traduit en format intermédiaire (IF). La spécification de format intermédiaire présente un protocole comme un ensemble de règles de réécriture.

Les règles de réécriture décrivent un système de transition à états infinis qui est appliqué afin de poursuivre le traitement d'un protocole donné par les outils Back-End de l'analyseur. De plus, il définit des règles de transition et une propriété de sécurité basée sur l'état. Un protocole, écrit en IF, est exécuté sur un nombre fini d'itérations, ou entièrement si aucune boucle n'est impliquée. Enfin, à la fin de l'analyse, soit une attaque est trouvée, soit le protocole est considéré comme sûr (sécurisé) pour le nombre donné de sessions. Dans le cas où le protocole n'est pas sécurisé, AVISPA fournit une trace qui met en évidence les étapes qui ont conduit à l'attaque.

Pour résumer, deux actions principales sont réalisées dans la simulation AVISPA :

1. Implémentation du protocole dans le langage HLPSL qui sera ensuite traduite à l'IF à l'aide du traducteur HLPSL2IF.
2. Produire le format de sortie OF à partir de l'IF fourni par l'un des Back-End, pour déterminer si le protocole est sûr.

Le format OF fournit deux sections imprimées appelées «SOMMAIRE» et «DÉTAILS». La section RÉSUMÉ détermine si le protocole est sûr ou non. La section DÉTAILS présente deux types de rapports :

1. Si le protocole est reconnu comme dangereux dans la section «RÉSUMÉ», la section «DÉTAILS» détermine certaines conditions qui doivent être respectées pour cette sécurité.
2. Si le résultat de la section «SUMMERY» est non-conclusif, sa cause est analysée.

À la fin de l'analyse, la trace de l'attaque (le cas échéant) est également possible au format Alice-Bob habituel.

6.4.2.2. Spécification du protocole

Les objectifs de sécurité du protocole TAKE-IoT sont spécifiés dans la section des objectifs (Goal Section) avant de lancer l'analyse. Afin d'évaluer notre protocole, nous avons effectué les étapes suivantes :

- **Étape 1:** cette première étape consiste à coder les rôles de base du protocole proposé. Donc, deux rôles de base sont définis dans le schéma proposé : «Alice»

et «Bob», représentant respectivement l'initiateur et le répondant. Le rôle de Bob dans le langage HLPSL est illustré à la figure 6.4.

```
role bob(B,A:agent, G: text,
          H: hash_func, SND_A, RCV_A: channel (dy))
played_by B
def=
  local State: nat,
  F1,F3,F4,F5,Add,Mul: function,
  Tti, Ttr, Vvi, Vvr,Kir: text,
  K,N,P: text,  SApr,SAch: text,  IDi, IDr: text,
  Li, Lr, Ei,Er: text, Ti, Tr,Vi,Vr: text,
  S1,S2,S3,S4: text, Z,W,U1,U2,X: text
  const  m1: text,
  sec_b_Kir : protocol_id

 init State := 1
 transition
2. State=1 /\ RCV_A (SApr'.IDi'.Ti')=|> State':=3
           /\ IDr':= new()
           /\ Vvr':=new()
           /\ SAch':=new()
           /\ Vr':= F1(Vvr',P)
           /\ Ttr':=new()
           /\ S1':=H(Ttr'.Vvr')
           /\ Tr':= F1(S1',P)
           /\ K':= F1(S1',Ti')
           /\ Er':= H(Tr')
           /\ S2':= exp(Vvr',m1)
           /\ S3':= F1(Tr',K')
           /\ S4':= Add(S2',S3')
           /\ Lr':= Mul(S2',S4')
     /\ Kir':= H(IDi'.IDr'.F3(Ti').F4(Tr').F5(K'))
     /\ SND_A (SAch'.Tr'.({IDi'.IDr'.Vr'.Lr'}_Kir'))
                /\ witness(A,B,sk2,K')

4. State= 3 /\ RCV_A ({SApr'.Vi'.Li'}_Kir)=|> State':=5
           /\ Z':= H(Ti)
           /\ W':= exp(Li',m1)
           /\ U1':= Z'.W'
           /\ U2':= K'.W'
           /\ X':= Add(U1'.P,U2'.Ti.P)
5. State=5 /\ (X'=Vi') =|> State':=7
           /\secret(Kir,sec_a_Kir,{A,B})/\ request
(A,B,kir1,Kir)
end role
```

Figure 6. 4 Le rôle de Bob

- **Etape 2** : Dans cette étape, la session, l'environnement et les rôles d'objectif sont définis dans le langage HLPSL. Le rôle environnement contient la connaissance initiale des intrus, les constantes globales et la composition d'une ou plusieurs sessions (Voir la Figure 6.5). Le rôle objectif sert à spécifier les dimensions de sécurité.
- **Etape 3 :** c'est la dernière étape, ou le protocole est vérifié et les résultats de la vérification sont obtenus.

```
role environement ()
def=
   const sk2,kir1,kir2:protocol_id,
       a,b,i:agent, g:text,
       snd,rcv:channel(dy),
       h :hash_func

intruder_knowledge = {a,b,i,g,h,snd,rcv}
composition

  session(a,b,g,h,snd, rcv) /\ session(a,b,g,h,snd, rcv)

end role
```

Figure 6. 5 Rôle de l'environnement

6.4.2.3. Analyse des résultats

TAKE-IoT est vérifié à l'aide de l'animateur de sécurité pour AVISPA (SPAN); les résultats de la vérification formelle obtenue par le Back-End OFMC sont présentés dans la figure 6.6. Cette vérification confirme que le protocole proposé est sûr et garantit la réalisation des propriétés de sécurité : sécurisation des données, authentification des parties et protection contre les attaques communes de WSN, telles que MITM, DOS, attaque par rejeu et l'attaque par réflexion.

```
%OFMC
%Version of 2006/02/13
SUMMARY
 SAFE
DETAILS
 BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
 C:\SPAN\testsuite\results\TAKE.if
GOAL
 As_specified
BACKEND
 OFMC
COMMENTS
STATISTICS
 parseTime: 0.00s
 searchTime: 0.68s
 visitedNodes: 208 nodes
 depth: 6 piles
```

Figure 6. 6 Résultats de la vérification formelle

6.4.3. Comparaison avec autres travaux relatifs

L'analyse théorique et formelle du protocole TAKE-IoT a montré qu'il est sécurisé et résistant à différents types d'attaques. Afin de confirmer son efficacité et son originalité, nous avons réalisé une étude comparative en termes de sécurité et de complexité entre les différents travaux relatifs.

6.4.3.1. Sécurité :

Concernant la sécurité, une étude comparative est menée entre TAKE-IoT et les protocoles pertinents proposés dans la littérature que nous avons présentés dans le chapitre IV. La comparaison est basée sur la satisfaction des propriétés de sécurité : PFS et Sécurité de la clé connue ainsi que la résistance aux attaques. Les résultats de cette étude comparative sont présentés dans le tableau 6.2. Visiblement, les versions antérieures d'IKEv2 se concentrent sur la façon de fournir les meilleurs mécanismes d'échange de clés. Cependant, la résistance contre les attaques courantes dans les réseaux de capteurs n'a pas retenu autant d'attention. En termes de sécurité, TAKE-IoT résiste aux attaques de Modification, Réflexion, Rejeu, DOS et MITM.

Par conséquent, il garantit plusieurs dimensions de sécurité comme l'authentification, l'intégrité et la confidentialité.

Travaux relatifs / Critères de comparaison	IKEv1 [61]	IKEv1 [62]	IKEv2 [104]	LKA [105]	TAKE-IoT
Perfect Forward Security	○	○	●	●	●
Sécurité de la clé connue	○	○	●	●	●
Attaque par Réflexion	×	×	✓	–	✓
Attaque par Rejeu	×	×	✓	–	✓
Attaque DOS	×	×	✓	–	✓
Attaque MITM	×	×	✓	×	✓

Tableau 6. 2 Etude comparative en termes de sécurité

● : « Satisfait »
○ : « Non-satisfait »
– : « non-précisé »
✓ : « Résistant »
× : « non-résistant »

6.4.3.2. Complexité

Nous avons effectué une analyse de la complexité ou nous avons comparé les performances du protocole proposé avec les autres versions du protocole IKE. Les critères utilisés dans cette analyse sont : la fonction pseudo-aléatoire qui présente le nombre de fonctions utilisées pour dériver le matériel de clé pour le cryptage et le décryptage des messages (C /D), la fonction de hachage utilisée pour C / D, le nombre de clés secrètes /publiques utilisées pour C /D, les opérations de calcul : Addition (Add), soustraction (Sou), multiplication (Mul), division (Div) et exponentielle (Exp). En plus

du nombre d'opérations XOR et le nombre de messages échangés. Les résultats de la comparaison sont présentés dans le tableau 6.3. En termes de complexité, TAKE-IoT utilise des opérations simples telles que l'addition et la division pour générer une clé sécurisée. En ce qui concerne les fonctions de hachage, TAKE-IoT augmente le nombre de fonctions de hachage par rapport à [105]. Cependant, cela offre plus de sécurité que leur proposition que nous avons démontré dans le chapitre VI qu'elle est vulnérable à l'attaque MITM. En ce qui concerne le nombre de messages, TAKE-IoT offre le nombre minimal de messages par rapport à tous les travaux cités.

Travaux relatifs / **Critères de comparaison**		**[61]**	**[62]**	**[104]**	**[105]**	**TAKE-IoT**
Fonction Pseudo-Aléatoire C/D		10/10	0/0	0/0	0/0	0/0
Fonction de Hachage C/D		0/0	2/2	2/2	2/2	3/3
Clés secrètes C/D		6/6	2/4	3/3	1/1	1/1
Clés publiques C/D		1/1	0/0	0/0	1/1	0/0
Opération XOR C/D		0/0	0/0	2/2	0/0	0/0
Operations de calculs C/D	Add	0/0	0/0	1/1	0/0	1/1
	Sou	0/0	0/0	1/1	0/0	0/0
	Div	0/0	0/0	0/0	0/0	2/2
	Mul	0/0	0/0	0/0	2/2	0/0
	Exp	2/2	5/6	0/0	0/0	0/0
Nombre de messages		6	4	4	4	3

Tableau 6. 3 Etude comparative en termes de complexité

IL est à noter que, Cette contribution a fait le sujet d'un article scientifique qui a été publié par le journal « International Journal of Embedded and Real-Time Communication Systems (IJERTCS) », Volume 11, Issue 3, Article 1. [121]

6.4.4. Discussion

Comme mentionné dans la section 6.1, l'objectif de cette contribution est de concevoir un protocole efficace d'échange de clés, à utiliser avec IPsec dans les systèmes IoT. Le protocole proposé utilise des opérations rapides, simples et légères, garantissant un faible coût de calcul, ce qui est essentiel pour les appareils IoT. De plus, TAKE-IoT réduit le nombre de messages échangés pour créer l'association de sécurité en échangeant seulement trois messages au lieu de quatre messages dans les versions conventionnelles d'IKEv2. L'utilisation d'ECC fournit le même niveau de sécurité avec moins de tailles de clés. A savoir, pour les clés secrètes de 128 bits, ECC utilise des clés de 256 bits au lieu de 3072 bits pour RSA. Par conséquent, TAKE-IoT peut fournir un haut niveau de sécurité tout en conservant les meilleures performances nécessaires pour les appareils 6LoWPAN. TAKE-IoT convient aux environnements où les transmissions de paquets sont périodiques ou dépendent de l'action de l'utilisateur. Par conséquent, les données seront transmises en toute sécurité entre le dispositif contraint, tel qu'un capteur de température ambiante, et un routeur 6LoWPAN.

6.5. Conclusion

Dans ce chapitre, nous avons présenté une solution basée IPsec qui est le protocole TAKE-IoT. TAKE-IoT est un protocole d'authentification et d'échange de clé pour les dispositifs 6LoWPAN, utilisés dans le contexte de l'Internet des Objets. Nous avons montré que notre solution est efficace en termes de calcul, car il repose sur la cryptographie légère. De plus, la solution proposée minimise le nombre de messages et utilise des opérations simples et légères. Nous avons adopté l'outil 'AVISPA' pour la vérification de la sécurité du protocole proposé. Les résultats de la vérification prouvent que TAKE-IoT est un protocole sécurisé, fiable et robuste contre les attaques de sécurité courantes dans les réseaux 6LoWPANs. TAKE-IoT vise à assurer les exigences de sécurité afin de résister aux différents types d'attaques. Enfin, nous avons comparé le protocole avec les solutions proposées dans la littérature. Nous avons démontré que TAKE-IoT surpasse les autres mécanismes en termes de sécurité, sans affecter les performances du réseau.

Conclusion générale

L'internet des objets basé sur IP est une technologie qui a éliminé les frontières entre le monde physique et le monde virtuel. L'IoT regroupe plusieurs domaines, c'est un système hétérogène. Ce paradigme utilise différentes technologies déployées sur des architectures et des plateformes implémentées sur des matériaux informatiques très variés. Désormais, l'IoT est intégrée dans notre vie quotidienne, la sécurité de ces systèmes est une tâche primordiale. Le protocole IPsec, indispensable pour le réseau internet, est un choix optimal pour l'IoT. Cependant, certaines adaptations doivent être apportées à ce protocole pour convenir aux exigences des systèmes IoT.

Dans cette thèse, nous avons étudié la possibilité d'exploiter le protocole IPsec dans les réseaux 6LoWPANs qui sont une technologie maîtresse qui a contribué au succès de l'IoT. Afin d'adapter le protocole IPsec aux réseaux 6LoWPANs, il est fortement recommandé d'utiliser des protocoles sous-jacents robustes et allégés. La sécurité du protocole IPsec dépend essentiellement de son protocole sous-jacent IKEv2. Bien que des efforts considérables aient faits pour adapter ce protocole à l'IoT, le protocole IKEv2 présente des faiblesses qui peuvent menacer la sécurité des communications dans les 6LoWPANs. Par conséquent, la motivation de cette thèse émerge du besoin de sécuriser les 6LoWPANs contre les diverses attaques communes dans ce type de réseau. Nous avons constaté que la solution optimale serait de concevoir des protocoles spécifiques à ces réseaux.

Contributions

Dans notre thèse, nous avons proposé deux améliorations principales du protocole IPsec afin de l'exploiter dans un environnement dynamique et hétérogène comme l'IoT.

La première proposition consiste à fournir un mécanisme efficace pour la gestion automatique des politiques de sécurité IPSec. Le but est de manipuler les politiques d'une façon automatique et rapide tout en minimisant l'intervention humaine. Les conflits des politiques de sécurité deviennent plus récurrents dans des environnements dynamiques et hétérogènes. Ainsi, nous avons proposé un mécanisme de détection et de résolution des conflits. Le but de cette proposition a été atteint. Nous avons proposé un algorithme de génération automatique des politiques de sécurité IPsec sans conflits

(AGCSP-IPsec). Nous avons démontré avec plusieurs expérimentations que l'algorithme AGCSP-IPsec est efficace en termes de temps de traitement et d'usage de mémoire. D'où, l'algorithme proposé est adapté aux changements fréquents des entités du réseau. L'algorithme AGCSP-IPsec est un moyen efficace pour remplacer la gestion manuelle des politiques. Par conséquent, cette proposition permet au protocole IPsec d'être plus adapté aux environnements dynamiques.

Notre deuxième proposition est la proposition d'un nouveau protocole d'authentification et d'échange de clé entre les dispositifs IoT. Cette proposition vise à concevoir un protocole qui peut être associé au protocole IPsec pour remplacer le protocole IKEv2, tout en restant compatible avec ce dernier. Le but de cette proposition a été atteint suite à la proposition du protocole TAKE-IoT, un protocole léger d'authentification et d'échange de clé. Nous avons concentré nos recherches pour adapter ce protocole aux contraintes des dispositifs IoT en termes de capacité de calcul et de mémoire ainsi que leur faible puissance. De ce fait, nous avons utilisé des techniques cryptographiques légères comme la cryptographie à courbe elliptique. En plus, nous avons minimisé le nombre d'échanges nécessaires pour établir une association de sécurité. En plus de l'allègement du protocole, nous avons amélioré la sécurité de notre proposition par rapport au protocole IKEv2. Le protocole proposé peut résister aux attaques les plus fréquentes dans les WSNs comme les attaques par dénis de service, les attaques par rejeu, l'attaque homme au milieu…, etc. la vérification de notre proposition a démontré que notre protocole pourrait non seulement améliorer la sécurité, mais également entraîner moins de frais de calcul et de transmission.

Perspectives

Les travaux réalisés dans le cadre de cette thèse ont permis d'atteindre l'objectif fixé initialement, qui est l'utilisation du protocole IPsec pour la sécurisation des communications de bout en bout dans les réseaux 6LoWPANs. Par ailleurs, plusieurs perspectives peuvent être considérées pour améliorer davantage notre travail. Comme travaux futurs, il est intéressant d'étendre l'algorithme AGSCP-IPsec à d'autres types de contrôle de sécurité. En outre, la stratégie de résolution des conflits inter-politiques peut

être améliorée. De plus, il est prévu d'étudier la consommation d'énergie et la complexité matérielle de l'approche proposée.

De plus, nous envisageons de concevoir une solution de gestion de clé de groupe sécurisée pour TAKE-IoT au sein d'une solution légère. Ainsi, tous les services seront possibles comme sur les appareils sans contraintes, pour permettre plus de fonctionnalités et améliorer la sécurité au sein des applications IoT.

Une autre perspective serait d'utiliser la technique de clustering pour améliorer la gestion de groupe de clé. Ceci va permettre la distribution d'implémentation d'IPsec sur plusieurs machines. On suppose que cette solution serait efficace pour les environnements qui exigent la disponibilité et les hautes performances. Cependant, cette distribution doit toutefois préserver la sécurité de l'IPsec et doit fournir une répartition dynamique des charges.

Références

Références

[1] Kushalnagar, N., & Montenegro, G. (2007). Transmission of IPv6 packets over IEEE 802.15. 4 networks.

[2] Lee, I., & Lee, K. (2015). The Internet of Things (IoT): Applications, investments, and challenges for enterprises. Business Horizons, 58(4), 431-440.

[3] Molisch, A. F., Balakrishnan, K., Chong, C. C., Emami, S., Fort, A., Karedal, J & Siwiak, K. (2004). IEEE 802.15. 4a channel model-final report. IEEE P802, 15 (04), 0662.

[4] Montenegro, G., Kushalnagar, N., Hui, J. and Culler, D. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, Internet Engineering Task Force, Sep. 2007.

[5] Hui, J. and Thubert, P. Compression Format for IPv6 Datagrams in 6LoWPAN Networks. Internet-Draft draft-ietf-6lowpan-hc-05, Internet Engineering Task Force, Jun. 2009.

[6] Nguyen, K. T., Laurent, M., & Oualha, N. (2015). Survey on secure communication protocols for the Internet of Things. Ad Hoc Networks, 32, 17-31.

[7] Benslimane, Y., Benahmed, K., & Benslimane, H. (2018, November). Security Mechanisms for 6LoWPAN Network in Context of Internet of Things: A Survey. In International Conference in Artificial Intelligence in Renewable Energetic Systems (pp. 49-69). Springer, Cham.

[8] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. Ad hoc networks, 1(2):293–315, 2003.

[9] O. Garcia-Morchon, R. Hummen, S.S. Kumar, R. Struik, and S.L. Keoh. Security Considerations in the IP-based Internet of Things, March 2012. http://tools.ietf.org/html/draft-garcia-core-security-04.

[10] Cervantes, C., Poplade, D., Nogueira, M., Santos A.: Detection of sinkhole attacks for supporting secure routing on 6LoWPAN for Internet of Things. In: 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 606–611. IEEE (2015).

[11] Pongle, P., Chavan, G.: Real time intrusion and wormhole attack detection in internet of things. Int. J. Comput. Appl. 121(9) (2015).

[12] Bostani, H., Sheikhan, M.: Hybrid of anomaly-based and specification-based IDS for Internet of Things using unsupervised OPF based on MapReduce approach. Comput. Commun. 98, 52– 71 (2017).

[13] Raza, S., Wallgren, L., Voigt, T.: SVELTE: real-time intrusion detection in the Internet of Things. Ad Hoc Netw. 11(8), 2661–2674 (2013b).

[14] Ibrahim Ethem Bagci, Mohammad Reza Pourmirza, Shahid Raza, UtzRoedig, and Thiemo Voigt. Codo: Confidential data storage for wireless sensor networks. In 8th IEEE International Workshop on Wireless and Sensor Networks Security (WSNS 2012), Las Vegas, Nevada, USA, October 2012.

[15] Sastry, N., & Wagner, D. (2004, October). Security considerations for IEEE 802.15. 4 networks. In Proceedings of the 3rd ACM workshop on Wireless security (pp. 32-42)

[16] Alharby, S., Weddell, A., Reeve, J., & Harris, N. (2018). The Cost of Link Layer Security in IoT Embedded Devices. IFAC-PapersOnLine, 51(6), 72-77.

[17] Raza, S., Duquennoy, S., Chung, T., Yazar, D., Voigt, T., & Roedig, U. (2011, June). Securing communication in 6LoWPAN with compressed IPsec. In 2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS) (pp. 1-8). IEEE.

[18] Raza, S., Duquennoy, S., & Selander, G. (2013). Compression of IPsec AH and ESP headers for constrained environments.

[19] Raza, S., Chung, T., Duquennoy, S., Voigt, T., & Roedig, U. (2010). Securing internet of things with lightweight ipsec.

[20] Wang, H., & Sun, Z. (2018). Compression Method for IPSec over 6LoWPAN. KSII Transactions on Internet & Information Systems, 12(4).

[21] Gurtov, A., Komu, M., & Moskowitz, R. (2009). Host Identity Protocol (HIP): Identifier/locator split for host mobility and multihoming. Internet Protocol Journal, 12(1), 27–32.

[22] Moskowitz, R. (2012). HIP Diet Exchange (DEX): draft-moskowitz-hip-dex-00. Standards Track.

[23] Hummen, R., Hiller, J., Henze, M., & Wehrle, K. (2013, October). Slimfit—A HIP DEX compression layer for the IP-based Internet of things. In 2013

IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob) (pp. 259-266). IEEE.

[24] Saied, Y. B., & Olivereau, A. (2012, March). HIP Tiny Exchange (TEX): A distributed key exchange scheme for HIP-based Internet of Things. In Third International Conference on Communications and Networking (pp. 1-8). IEEE.

[25] Lu, Y., & Da Xu, L. (2018). Internet of things (iot) cybersecurity research: A review of current research topics. IEEE Internet of Things Journal, 6(2), 2103-2115.

[26] Glissa, G., & Meddeb, A. (2019). 6LowPSec: An end-to-end security protocol for 6LoWPAN. Ad Hoc Networks, 82, 100-112.

[27] Rescorla, E., & Modadugu, N. (2006). Datagram transport layer security.

[28] Raza, S., Shafagh, H., Hewage, K., Hummen, R., & Voigt, T. (2013). Lithe: Lightweight secure CoAP for the internet of things. IEEE Sensors Journal, 13(10), 3711-3720.

[29] Kothmayr, T., Schmitt, C., Hu, W., Brünig, M., & Carle, G. (2012, October). A DTLS based end-to-end security architecture for the Internet of Things with two-way authentication. In 37th Annual IEEE Conference on Local Computer Networks-Workshops (pp. 956-963). IEEE.

[30] Granjal, J., Monteiro, E., & Silva, J. S. (2013, May). End-to-end transport-layer security for Internet-integrated sensing applications with mutual and delegated ECC public-key authentication. In 2013 IFIP Networking Conference (pp. 1-9). IEEE.

[31] Chavan, A. A., & Nighot, M. K. (2016). Secure and cost-effective application layer protocol with authentication interoperability for IOT. Procedia Computer Science, 78, 646-651.

[32] Park, C. S., & Park, W. S. (2018). A Group-Oriented DTLS Handshake for Secure IoT Applications. IEEE Transactions on Automation Science and Engineering, 15(4), 1920-1929.

[33] Z. Shelby, K. Hartke, and C. Bormann. Constrained Application Protocol (CoAP), March 2013. http://tools.ietf.org/html/draft-ietf-core-coap-14.

[34] Bormann, C., Castellani, A. P., & Shelby, Z. (2012). Coap: An application protocol for billions of tiny internet nodes. IEEE Internet Computing, 16(2), 62-67.

[35] Granjal, J., Monteiro, E., & Silva, J. S. (2013, June). Application-layer security for the WoT: Extending CoAP to support end-to-end message security for Internet-integrated sensing applications. In International Conference on Wired/Wireless Internet Communication (pp. 140-153). Springer, Berlin, Heidelberg.

[36] Korzun, D. G., Borodin, A. V., Paramonov, I. V., Vasilyev, A. M., & Balandin, S. I. (2015). Smart spaces enabled mobile healthcare services in internet of things environments. International Journal of Embedded and Real-Time Communication Systems (IJERTCS), 6(1), 1-27.

[37] Snader, J. C. (2015). VPNs Illustrated: Tunnels, VPNs, and IPsec. Addison-Wesley Professional.

[38] Kent, S., & Atkinson, R. (1998). RFC 2402: IP authentication header (AH). IETF, [online] https://tools.ietf.org/html/rfc2402, [consulté en Décembre 2016].

[39] Kent, S., & Atkinson, R. (1998). RFC: 2406: IP encapsulating security payload (ESP). IETF [online] https://tools.ietf.org/html/rfc24026, [consulté en Janvier 2017]

[40] Lang, U., & Schreiner, R. (2015). U.S. Patent No. 9,043,861. Washington, DC: U.S. Patent and Trademark Office.

[41] Maughan, D., Schertler, M., Schneider, M., & Turner, J. (1998). Internet security association and key management protocol (ISAKMP).

[42] Maughan, D., Schertler, M., Schneider, M., & Turner, J. (1998). Internet security association and key management protocol (ISAKMP).

[43] Pisharody, S., Chowdhary, A., & Huang, D. (2016, October). Security policy checking in distributed SDN based clouds. In 2016 IEEE Conference on Communications and Network Security (CNS) (pp. 19-27). IEEE.

[44] Wu, Z., & Liu, Y. (2013). Knowledge-based policy conflict analysis in mobile social networks. Wireless personal communications, 73(1), 5-22.

[45] Fu, Z., Wu, S. F., Huang, H., Loh, K., Gong, F., Baldine, I., & Xu, C. (2001, January). IPSec/VPN security policy: Correctness, conflict detection, and

resolution. In International Workshop on Policies for Distributed Systems and Networks (pp. 39-56). Springer, Berlin, Heidelberg.

[46] LI, Zhitang, CUI, Xue, et CHEN, Lin. Analysis and classification of ipsec security policy conflicts. In : Frontier of Computer Science and Technology, 2006. FCST'06. Japan-China Joint Workshop on. IEEE, 2006. p. 83-88.

[47] HAMED, Hazem et AL-SHAER, Ehab. Taxonomy of conflicts in network security policies. IEEE Communications Magazine, 2006, vol. 44, no 3, p. 134-141.

[48] Al-Shaer, E. (2014). Modeling and verification of firewall and ipsec policies using binary decision diagrams. In Automated Firewall Analytics (pp. 25-48). Springer, Cham.

[49] Hamed, H., Al-Shaer, E., & Marrero, W. (2005, November). Modeling and verification of IPSec and VPN security policies. In 13th IEEE International Conference on Network Protocols (ICNP'05) (pp. 10-pp). IEEE.

[50] Sun, H. M., Chang, S. Y., Chen, Y. H., He, B. Z., & Chen, C. K. (2007). The design and implementation of IPSec conflict avoiding and recovering system. In TENCON 2007-2007 IEEE Region 10 Conference (pp. 1-4). IEEE.

[51] Niksefat, S., & Sabaei, M. (2010, April). Efficient algorithms for dynamic detection and resolution of IPSec/VPN security policy conflicts. In 2010 24th IEEE International Conference on Advanced Information Networking and Applications (pp. 737-744). IEEE.

[52] Al-Shaer, E. S., & Hamed, H. H. (2003, March). Firewall policy advisor for anomaly discovery and rule editing. In International Symposium on Integrated Network Management (pp. 17-30). Springer, Boston, MA.

[53] Yuan, L., Chen, H., Mai, J., Chuah, C. N., Su, Z., & Mohapatra, P. (2006, May). Fireman: A toolkit for firewall modeling and analysis. In 2006 IEEE Symposium on Security and Privacy (S&P'06) (pp. 15-pp). IEEE.

[54] Hu, H., Han, W., Ahn, G. J., & Zhao, Z. (2014, August). FLOWGUARD: building robust firewalls for software-defined networks. In Proceedings of the third workshop on hot topics in software defined networking (pp. 97-102). ACM.

[55] Ferraresi, S., Pesic, S., Trazza, L., & Baiocchi, A. (2007, June). Automatic conflict analysis and resolution of traffic filtering policy for firewall and

security gateway. In 2007 IEEE International Conference on Communications (pp. 1304-1310). IEEE.

[56] Basile, C., Cappadonia, A., & Lioy, A. (2012). Network-level access control policy analysis and transformation. IEEE/ACM Transactions on Networking (TON), 20(4), 985-998.

[57] Bouhoula, A., & Yazidi, A. (2016, October). A security policy query engine for fully automated resolution of anomalies in firewall configurations. In 2016 IEEE 15th International Symposium on Network Computing and Applications (NCA) (pp. 76-80). IEEE.

[58] Singh, S. P., Bharti, V., Singh, B. K., Johri, P., & Sharma, M. (2016, April). Formation of security association database (SAD) in internet protocol version 6 (IPv6). In 2016 International Conference on Computing, Communication and Automation (ICCCA) (pp. 491-497). IEEE.

[59] Harkins, D., & Carrel, D. (1998). The internet key exchange (IKE). RFC 2409, november.

[60] Kaufman, C. (2005). Internet key exchange (IKEv2) protocol (pp. 1-37). RFC 4306, December.

[61] Cheng, P. C. (2001). An architecture for the Internet Key Exchange protocol. *IBM Systems Journal*, *40*(3), 721-746.

[62] Su, M. Y., & Chang, J. F. (2007, May). An efficient and secured internet key exchange protocol design. In Fifth Annual Conference on Communication Networks and Services Research (CNSR'07) (pp. 184-192). IEEE.

[63] Iso-Anttila, L., Ylinen, J., & Loula, P. (2007, October). A proposal to improve IKEv2 negotiation. In The International Conference on Emerging Security Information, Systems, and Technologies (SECUREWARE 2007) (pp. 169-174). IEEE.

[64] Perlman, R., & Kaufman, C. (2000). Key exchange in IPSec: Analysis of IKE. IEEE Internet Computing, 4(6), 50-56.

[65] Diffie, W., & Hellman, M. E. (1976, June). Multiuser cryptographic techniques. In Proceedings of the June 7-10, 1976, national computer conference and exposition (pp. 109-112).

[66] Andress, J. (2014). The basics of information security: understanding the fundamentals of InfoSec in theory and practice. Syngress.

[67] Raza, S., Seitz, L., Sitenkov, D., & Selander, G. (2016). S3K: Scalable security with symmetric keys—DTLS key establishment for the Internet of Things. IEEE Transactions on Automation Science and Engineering, 13(3), 1270-1280.

[68] Chandramouli, R., Bapatla, S., Subbalakshmi, K. P., & Uma, R. N. (2006). Battery power-aware encryption. ACM Transactions on Information and System Security (TISSEC), 9(2), 162-180.

[69] Surendran, S., Nassef, A., & Beheshti, B. D. (2018, May). A survey of cryptographic algorithms for IoT devices. In 2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT) (pp. 1-8). IEEE.

[70] Usman, M., Ahmed, I., Aslam, M. I., Khan, S., & Shah, U. A. (2017). SIT: a lightweight encryption algorithm for secure internet of things. arXiv preprint arXiv:1704.08688.

[71] Eschenauer, L., & Gligor, V. D. (2002, November). A key-management scheme for distributed sensor networks. In Proceedings of the 9th ACM conference on Computer and communications security (pp. 41-47).

[72] Sciancalepore, S., Piro, G., Boggia, G., & Bianchi, G. (2016). Public key authentication and key agreement in IoT devices with minimal airtime consumption. IEEE Embedded Systems Letters, 9(1), 1-4.

[73] Saeed, M. E. S., Liu, Q. Y., Tian, G., Gao, B., & Li, F. (2019). AKAIoTs: Authenticated key agreement for Internet of Things. Wireless Networks, 25(6), 3081-3101.

[74] Aumasson J P, Neves S, Wilcox-OHearn Z and Winnerlein C 2013 BLAKE2: simpler, smaller, fast as MD5. In: Proceedings of the International Conference on Applied Cryptography and Network Security, pp. 119–135.

[75] Ouellet, E., & O'Farrell, N. (2002). Hackproofing Your Wireless Network. Syngress Publishing.

[76] Coppersmith, D. (1994). The Data Encryption Standard (DES) and its strength against attacks. IBM journal of research and development, 38(3), 243-250.

[77] Daemen, J., & Rijmen, V. (2013). The design of Rijndael: AES-the advanced encryption standard. Springer Science & Business Media.

[78] Bao, F., Deng, R. H., & Zhu, H. (2003, October). Variations of diffie-hellman problem. In International conference on information and communications security (pp. 301-312). Springer, Berlin, Heidelberg.

[79] Mahajan, P., & Sachdeva, A. (2013). A study of encryption algorithms AES, DES and RSA for security. Global Journal of Computer Science and Technology.

[80] Koblitz, N. (1987). Elliptic curve cryptosystems. Mathematics of computation, 48(177), 203-209.

[81] Johnson, D., Menezes, A., & Vanstone, S. (2001). The elliptic curve digital signature algorithm (ECDSA). International journal of information security, 1(1), 36-63.

[82] D. Kravitz, "Digital signature algorithm," jul 1993, uS Patent 5,231,668.

[83] Gentry, C. (2003, May). Certificate-based encryption and the certificate revocation problem. In International Conference on the Theory and Applications of Cryptographic Techniques (pp. 272-293). Springer, Berlin, Heidelberg.

[84] Hammi, M. T., Hammi, B., Bellot, P., & Serhrouchni, A. (2018). Bubbles of Trust: A decentralized blockchain-based authentication system for IoT. Computers & Security, 78, 126-142.

[85] Yao, X., Han, X., Du, X., & Zhou, X. (2013). A lightweight multicast authentication mechanism for small scale IoT applications. IEEE Sensors Journal, 13(10), 3693-3701.

[86] Petrov, V., Edelev, S., Komar, M., & Koucheryavy, Y. (2014, March). Towards the era of wireless keys: How the IoT can change authentication paradigm. In 2014 IEEE World Forum on Internet of Things (WF-IoT) (pp. 51-56). IEEE.

[87] Kim, Y. P., Yoo, S., & Yoo, C. (2015, January). DAoT: Dynamic and energy-aware authentication for smart home appliances in Internet of Things. In 2015 IEEE International Conference on Consumer Electronics (ICCE) (pp. 196-197). IEEE.

[88] Aman, M. N., Chua, K. C., & Sikdar, B. (2017). Mutual authentication in IoT systems using physical unclonable functions. IEEE Internet of Things Journal, 4(5), 1327-1340.

[89] Hsu, C. H., Wang, S., Zhang, D., Chu, H. C., & Lu, N. (2016). Efficient identity authentication and encryption technique for high throughput RFID system. Security and Communication Networks, 9(15), 2581-2591.

[90] Gaikwad, P. P., Gabhane, J. P., & Golait, S. S. (2015, September). 3-level secure Kerberos authentication for Smart Home Systems using IoT. In 2015 1st International Conference on Next Generation Computing Technologies (NGCT) (pp. 262-268). IEEE.

[91] Panwar, M., & Kumar, A. (2015, March). Security for IoT: An effective DTLS with public certificates. In 2015 International Conference on Advances in Computer Engineering and Applications (pp. 163-166). IEEE.

[92] Park, A. J., Kim, H. Y., & Lim, J. I. (2015, August). A framework of device authentication management in IoT environments. In 2015 5th International Conference on IT Convergence and Security (ICITCS) (pp. 1-3). IEEE.

[93] Porambage, P., Schmitt, C., Kumar, P., Gurtov, A., & Ylianttila, M. (2014, April). Two-phase authentication protocol for wireless sensor networks in distributed IoT applications. In 2014 IEEE Wireless Communications and Networking Conference (WCNC) (pp. 2728-2733). Ieee.

[94] Zhao, G., Si, X., Wang, J., Long, X., & Hu, T. (2011, June). A novel mutual authentication scheme for Internet of Things. In Proceedings of 2011 International Conference on Modelling, Identification and Control (pp. 563-566). IEEE.

[95] Li, N., Liu, D., & Nepal, S. (2017). Lightweight mutual authentication for IoT and its applications. IEEE Transactions on Sustainable Computing, 2(4), 359-370.],

[96] Naoui, S., Elhdhili, M. E., & Saidane, L. A. (2016, November). Security analysis of existing IoT key management protocols. In 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA) (pp. 1-7). IEEE.

[97] Eschenauer, L., & Gligor, V. D. (2002, November). A key-management scheme for distributed sensor networks. In Proceedings of the 9th ACM conference on Computer and communications security (pp. 41-47).

[98] Basu, S., & Pushpalatha, M. (2013, December). Analysis of energy efficient ECC and TinySec based security schemes in Wireless Sensor Networks. In

2013 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS) (pp. 1-6). IEEE.

[99] Rabiah, A. B., Ramakrishnan, K. K., Liri, E., & Kar, K. (2018, February). A lightweight authentication and key exchange protocol for IoT. In Proc. Workshop Decentralized IoT Secur. Standards (DISS).

[100] Saied, Y. B., Olivereau, A., Zeghlache, D., & Laurent, M. (2014). Lightweight collaborative key establishment scheme for the Internet of Things. Computer Networks, 64, 273-295.

[101] Garcia-Morchon, O., Rietman, R., Sharma, S., Tolhuizen, L., & Torre-Arce, J. L. (2015, September). DTLS-HIMMO: Achieving DTLS certificate security with symmetric key overhead. In European Symposium on Research in Computer Security (pp. 224-242). Springer, Cham.

[102] Ben Saied, Y., & Olivereau, A. (2012, October). (k, n) threshold distributed key exchange for HIP based internet of things. In Proceedings of the 10th ACM international symposium on Mobility management and wireless access (pp. 79-86).

[103] Raza, S., Voigt, T., & Jutvik, V. (2012, March). Lightweight IKEv2: a key management solution for both the compressed IPsec and the IEEE 802.15. 4 security. In Proceedings of the IETF workshop on smart object security (Vol. 23).

[104] Ahmim, M., Babes, M., & Ghoualmi-Zine, N. (2015). Formal analysis of efficiency and safety in IPSec based on internet key exchange protocol. International Journal of Communication Networks and Distributed Systems, 14(2), 202-218.

[105] Lavanya, M., & Natarajan, V. (2017). Lightweight key agreement protocol for IoT based on IKEv2. Computers & Electrical Engineering, 64, 580-594.

[106] Kivinen, T. (2016). Minimal Internet Key Exchange Version 2 (IKEv2) Initiator Implementation.

[107] Raza, S., & Magnússon, R. M. (2018). TinyIKE: Lightweight IKEv2 for Internet of Things. IEEE Internet of Things Journal, 6(1), 856-866.

[108] Abbes, T., Bouhoula, A., & Rusinowitch, M. (2016). Detection of firewall configuration errors with updatable tree. International Journal of Information Security, 15(3), 301-317.

[109] Korzun, D. G., Borodin, A. V., Paramonov, I. V., Vasilyev, A. M., & Balandin, S. I. (2015). Smart spaces enabled mobile healthcare services in internet of things environments. International Journal of Embedded and Real-Time Communication Systems (IJERTCS), 6(1), 1-27.

[110] Langhammer, N., & Kays, R. (2012). Performance evaluation of wireless home automation networks in indoor scenarios. IEEE Transactions on Smart Grid, 3(4), 2252-2261.

[111] Hussian, R., Sharma, S., Sharma, V., & Sharma, S. (2013). WSN applications: Automated intelligent traffic control system using sensors. Int. J. Soft Comput. Eng, 3(3), 77-81.

[112] Naor M and Yung M 1989 Universal one-way hash functions and their cryptographic applications. In: Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing ACM Press, pp. 33–43.

[113] Sobti R and Geetha G 2015 Performance comparison of Keccak, Skein, Grstl, Blake and JH: SHA-3 final round candidate algorithms on ARM Cortex A8 Processor. Int. J. Security Appl. 9(12): pp. 367–384.

[114] Lavanya, M., & Natarajan, V. (2017). LWDSA: light-weight digital signature algorithm for wireless sensor networks. Sādhanā, 42(10), 1629-1643.

[115] Aumasson, J. P., Henzen, L., Meier, W., & Phan, R. C. W. (2010). SHA-3 proposal BLAKE. Submission to the NIST SHA-3 Competition (Round 3).

[116] Armando, A., Basin, D., Cuellar, J., Rusinowitch, M., & Viganò, L. (2006). Avispa: automated validation of internet security protocols and applications. ERCIM News, 64(January).

[117] Khelf, R., & Ghoualmi-Zine, N. (2019, Mars). A Survey on Dynamic Multipoint Virtual Private Networks. In: the 8th International Seminar on Computer Science Research at Feminine (RIF 2019), (2379), 15-24.

[118] Khelf, R., & Ghoualmi-Zine, N. (2018, November). Ipsec/firewall security policy analysis: A survey. In 2018 International Conference on Signal, Image, Vision and their Applications (SIVA) (pp. 1-7). IEEE.

[119] Khelf, R., & Ghoualmi-Zine, N. AGCSP-IPsec: An Efficient Algorithm for Automatic Generation of Conflict-Free Security Policy of IPsec, International Journal of Information Security and Privacy (IJISP) (2020). [Article sous presse].

[120] Khelf, R., & Ghoualmi, N. (2017, February). Intra and inter policy Conflicts Dynamic Detection Algorithm. In 2017 Seminar on Detection Systems Architectures and Technologies (DAT) (pp. 1-6). IEEE.

[121] Khelf, R., Ghoualmi-Zine, N., & Ahmim, M. (2020). TAKE-IoT: Tiny Authenticated Key Exchange Protocol for the Internet of Things. International Journal of Embedded and Real-Time Communication Systems (IJERTCS), 11(3), 1-21.

Annexes A.1.

Liste des publications

Liste des publications

Articles de journal

[J1] Khelf, Roumaissa, Nacira Ghoualmi-Zine, and Marwa Ahmim. "TAKE-IoT: Tiny Authenticated Key Exchange Protocol for the Internet of Things." International Journal of Embedded and Real-Time Communication Systems (IJERTCS) volume 11, issue 3, P. 1-21. (2020).

[J2] KHELF, Roumaissa, GHOUALMI-ZINE, Nacira, AGCSP-IPsec: An Efficient Algorithm for Automatic Generation of Conflict-Free Security Policy of IPsec, International Journal of Information Security and Privacy (IJISP) (2020). Article sous presse.

Conférences internationales

[C1] KHELF, Roumaissa et GHOUALMI, Nassira. Intra and inter policy Conflicts Dynamic Detection Algorithm. In: 2017 Seminar on Detection Systems Architectures and Technologies (DAT). IEEE. p. 1-6 (2017).

[C2] KHELF, Roumaissa et GHOUALMI-ZINE, Nacira. IPsec/Firewall Security Policy Analysis: A Survey. In: 2018 International Conference on Signal, Image, Vision and their Applications (SIVA). IEEE, p. 1-7 (2018).

[C3] KHELF, Roumaissa et GHOUALMI-ZINE, Nacira. A Survey on Dynamic Multipoint Virtual Private Networks. In: the 8th International Seminar on Computer Science Research at Feminine (RIF 2019), CEUR-WS volume: 2379, p. 15-24. (2019).

Conférences nationales

[C4] KAHYA, Noudjoud, GHOUALMI, Nacira, LAFOURCADE, Pascal, KHELF Roumaissa, Formal Analysis of Key Management in mobile Wimax (2016).

Doctoriales

[D1] Khelf Roumaissa, Nacira Ghoualmi, “Analysis of IPsec security policies conflicts”. TUNISO-Algerian Doctoral Symposium in Computer Science (TADSCS'2017), Hammamet, Tunisia 24-26 April (2017).

[D2] Khelf Roumaissa, Nacira Ghoualmi, « Dynamic Verification of IPsec Security Policy: Conflicts Management». Journée doctoral en informatique (JDI’2017), Département d’informatique, Université Badji-Mokhtar Annaba, Algérie 12 décembre 2017

Annexes A.2.
Glossaire

Annexe A.2. Glossaire

3DES	Triple Data Encryption Algorithm
6LBR	6LoWPAN Border Router
6LoWPAN	IPv6 Low power Wireless Personal Area Networks
6LR	6LoWPAN router
ACL	Access Control List
AES	Advanced Encryption Standard
AGCSP-IPsec	Automatic Generation of Conflict-free Security Policy of IPsec
AH	Authentication Header
AVISPA	Automated Validation of Internet Security Protocols and Applications
CoAP	Constrained Application Protocol
DES	Data Encryption Algorithm
DH	Diffie-Hellman
DHP	Diffie-Hellman Problem
DLP	Discrete Logarithm Problem
DoS	Denial of Service
DTLS	Datagram Transport Layer Security
E2E	End to End
E2E	End to End
ECC	Elliptic Curve Cryptography
ECDHA	Elliptic curve Diffie–Hellman Algorithm
EL	Encryption List
ESP	Encapsulating Security Payload
FDD	Full Function Device
FMR	First Matching Rule
HIP	Host Identity Protocol
HLPSL	High Level Protocol Specification Language
HMAC	keyed-hash message authentication code
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers

IETF	Internet Engineering Task Force
IKE	Internet Key Exchange
IoT	Internet of Things
IPHC	IP Header Compression
IPv6	Internet Protocol version 6
KMP	Key Management Protocol
M2M	Machine to Machine
MD5	Message-Digest 5
MITM	Man In The Middle
NHC	Next Header Compression
NIST	National Institute of Standards and Technology
OBDD	Ordered Binary Decision Diagram
OSI	Open Systems Interconnection
PAD	Peer Authorization Database
PFS	Perfect Forward Secrecy
PKI	Public Key Infrastructure
RFD	Reduced Function Device
RSA	Rivest, Shamir and Adleman
SA	Security Association
SAD	Security Association Database
SHA	Security Hash Algorithm
SPD	Security Policy Database
SPI	Security Policy Identifier
SSL	Secure Sockets Layer
TAKE-IoT	Tiny Authenticated Key exchange Protocol for the Internet of things
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
VPN	Virtual Private Network
WSN	Wireless Sensor Network