REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEURE ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE IBN KHALDOUN - TIARET

MEMOIRE

Présenté à :

FACULTÉ MATHEMATIQUES ET INFORMATIQUE
DÉPARTEMENT D'INFORMATIQUE

Pour l'obtention du diplôme de :

MASTER

Spécialité : Réseaux et Télécommunication

Par :

❖ BAROUDA Mustapha
❖ ABDI Dieb Abdelghafour

Sur le thème

La détection des attaques DoS en utilisant l'analyse de la variance ANOVA dans les réseaux de capteurs sans fils

---

Soutenu publiquement le 24 / 06 / 2018 à Tiaret devant le jury composé de :

Mr MOKHTARI Ahmed	Grade	Université MAA	Président
Mr BOUALEM Adda	Grade	Université MAA	Examinateur
Mr BEKKAR Khaled	Grade	Université MAA	Encadreur

Remerciements

Nous remercions avant tout ALLAH qui nous a permis d'arriver jusque là

Paix et salut sur notre premier éducateur le prophète

Nous tenons à présenter respectueusement notre sincère gratitude à

MONSIEUR / BEEKAR KHALED notre encadreur qui nous a

Aider à réaliser ce travail en nous prodiguant des conseils, des orientations et des

observations aussi bénéfiques que fructueuses les unes que les autres.

Nous tenons aussi à remercier les jurés d'avoir accepté de juger notre travail et de

nous honorer de leur présence.

Dédicace

C'est avec profonde gratitude et sincères mots, que nous dédions ce modeste travail de fin d'étude à nos chers parents ; qui ont sacrifié leur vie pour notre réussite et nous ont éclairé le chemin par leurs conseils judicieux.

Nous espérons qu'un jour, nous pourrons leurs rendre un peu de ce qu'ils ont fait pour nous, que dieu leur prête bonheur et longue vie.

Nous dédions aussi ce travail à nos frères et sœurs, nos familles, nos amis, tous nos professeurs qui nous ont enseigné et à tous ceux qui nous sont chers.

Résumé

Le domaine d'application des réseaux de capteurs sans fil (RCSF) ne cesse d'accroître avec le besoin d'un mécanisme de sécurité efficace. Le fait que les réseaux de capteurs sans fil traitent des données très souvent sensibles, opérant dans des environnements hostiles et inattendus, les réseaux de capteurs sans fil (RCSF) peuvent être victimes de plusieurs attaques. L'attaque déni de service(DoS) est l'une des plus dangereuses, car ils peuvent avoir des impacts négatifs sur les applications critiques des RCSF, donc la notion de sécurité est considérée comme indispensable. Cependant, à cause de la limitation des ressources et la faible capacité de calcul d'un nœud capteur, le développement d'un mécanisme garantissant une sécurité pose de vrais défis de conception. Parmi les solutions proposées, les systèmes de détection d'intrusion (IDS) basés sur les modèles statistiques ont prouvé leur efficacité.

Dans ce travail, nous avons étudié les possibilités de détecter les attaques DoS en analysant la variance (ANOVA) entre le profit normal et les futures valeurs des attributs qu'on a choisi sur la base de leur impact sur le fonctionnement normal d'un RCSF.

Nous avons simulé quelques attaques : Blackhole, Hello-Flood et DoS, et sur la base de 100 simulations déployées, les performances de notre système sont plutôt acceptables, nous avons atteint 70% de précision dans le scénario de Blackhole, et 61% de précision dans le scenario Hello-Flood, et 65% de précision dans le scenario DoS.

Mots-clés: Réseaux de capteurs sans fil (RCSF), Système de détection d'intrusion (IDS), Simulations, fonctionnalités, déni de service (DoS), Blackhole, Hello-Flood, ANOVA.

Sommaire

Liste des figures

Liste des tableaux

Liste des abréviations

A

ANOVA: ANalysis Of Variance

ADC : Analog to Digital Converters

D

DoS : Denial of Service, « déni de service » en français

E

EAR: Eavesdrop And Register

H

H-IDS: Host Based Intrusion Detection System

I

IDS: Intrusion Detection System

L

LEACH : Low-Energy Adaptive Clustering Hierarchy

N

N-IDS: Network Based Intrusion Detection System

R

RCSF: Réseau de Capteur Sans Fil

RC: Rayon de Communication

RS: Rayon de Sensation

S

SMP: Sensor Management Protocol

SAR: Sequential Assignement Routing

SMACS: Self-organizing Medium Access Control for Sensor networks

T

TADAP: Task Assignement and Data Advertisement Protocol

TCP: Transmission Control Protocol

U

UDP: User Datagram Protocol

Introduction Générale :

Les facilités de déploiement des capteurs sans fil et la baisse de leur coût ont permis de généraliser l'utilisation de réseaux de capteurs sans fil (RCSF). Aujourd'hui, on retrouve ce type de réseau aussi bien dans la surveillance industrielle, que dans la mesure de données environnementales [1] [2], la domotique, la détection d'incendie [7], le milieu médical [3] ou bien encore dans le domaine militaire. La plupart de ces applications ont pour mission de surveiller une zone et d'obtenir une réaction quand elles détectent une donnée critique.

La divulgation de ces données critiques peut ne pas avoir une grande incidence dans des domaines tels que la domotique ou bien la capture d'événement environnemental. Sa confidentialité peut par contre être indispensable dans d'autres applications, comme pour le secret médical d'un patient à l'hôpital ou pour la sécurité du territoire dans le domaine militaire.

Un exemple de ces applications critiques est décrit dans le projet CodeBlue [3], où des capteurs recueillent des informations d'un patient. D'autres exemples existent aussi dans les applications militaires, par exemple la surveillance d'une zone de guerre ou l'enregistrement de l'état de santé ou de la position des troupes.

Dans ces deux cas, la confidentialité de l'information est primordiale, d'un point de vue juridique dans le premier cas, et d'un point de vue sécuritaire dans le second. Cette sécurité est bien sûr mise en danger par le médium utilisé, à savoir les ondes, mais également par les vulnérabilités spécifiques aux RCSF. Les solutions utilisées dans les réseaux ad hoc classiques ne peuvent pas s'appliquer stricto sensu aux RCSF, car ces dispositifs sont limités par leur batterie et leur puissance de calcul.

Concrètement, les solutions de cryptographies utilisées actuellement, comme les clés asymétriques, sont des solutions trop lourdes pour être calculées par les processeurs des capteurs actuels. Les protocoles de sécurisation doivent aussi limiter le nombre de messages nécessaires à leur bon fonctionnement, car la communication entre capteurs est la principale source de consommation d'énergie.

Ces contraintes [4] nous obligent actuellement à repenser à des solutions efficaces en termes de rapidité de calcul et de consommation énergétique, pour pouvoir sécuriser les réseaux de capteurs sans fil sans consommer leur énergie.

Les mécanismes de détection d'intrusion peuvent être des solutions beaucoup plus adaptées aux contraintes citées en dessus. En plus de dissuader l'intrus de paralyser le réseau, le système de détection d'intrusion (IDS) peut identifier des attaques connues ou inconnues [5].
Les IDS sont classés en trois catégories en fonction de leur méthodologie de détection : méthode basée sur l'abus, basée sur la spécification et basée sur l'anomalie [6]. Ce dernier a plusieurs techniques de détection : basée sur la connaissance, basée sur l'apprentissage automatique, et la détection basée sur la statistique. Ce système de détection présente un avantage par rapport au précèdent il détecte les nouveaux types d'attaques. Cependant il faudra faire parfois des ajustements afin que le fonctionnement de référence corresponde au mieux à l'activité normale des utilisateurs et ainsi réduire les fausses alertes qui en découleraient.
Notre problématique et de détecter des attaques DoS à travers un IDS basé sur un modèle statistique qui est l'analyse de la variance (ANOVA). Notre objectif est d'étudier l'influence des attaques DoS sur la variance d'un ou plusieurs attributs d'un réseau de capteur sans fil.
Le document est structuré comme suit :
- Dans le chapitre 1, nous traiterons les concepts des RCSF, nœuds de capteurs et de leurs composants logiciels et matériels, des caractéristiques principales d'un capteur, quelques applications et l'architecture et les systèmes d'exploitation des réseaux de capteurs.
- Dans le chapitre 2, nous allons expliquer l'intrusion et leurs types. Ensuite, nous décrirons les méthodes de détections tout en nous concentrant sur les IDS basés sur les anomalies, puis nous présentant les attaques les plus communes dans les RCSF.
- Au chapitre 3, nous allons expliquer notre approche, les outils que nous avons utilisés, expériences et commentaires sur les résultats.
- Les codes et scripts liés aux simulations et l'extraction des caractéristiques est rapportée aux annexes A et B et C respectivement.

Chapitre 01

1. Introduction :

Les avancées récentes en technologie des systèmes micro-électromécanique, des communications sans fil, et de l'électronique numérique ont permis le développement des nœuds capteurs peu coûteux, multifonctionnels et de basse puissance. Ces capteurs sont petits par la taille et communiquent sur des courtes distances. Ces nœuds capteurs minuscules sont constitués de composants de capture, de traitement de données et de communications. Ils ont influencé l'idée des réseaux de capteurs basés sur la collaboration d'un grand nombre des nœuds. Chaque nœud est un dispositif électronique qui possède une capacité de calcul, de stockage, de communication et d'énergie. Chaque capteur est doté d'un module d'acquisition qui lui permet de mesurer des informations environnementales : température, pression, accélération, sons, image, vidéo...etc. L'étendue des applications des réseaux de capteurs est vaste, on les retrouve dans le domaine de la santé, de la sécurité et dans le secteur militaire. Les réseaux de capteurs permettent à l'utilisateur une meilleure compréhension de l'environnement. De nos jours, les réseaux de capteurs sans fils font partie intégrante de notre vie.

2. Historique des réseaux de capteurs sans fil :

Les récents progrès des nouvelles techniques ont provoqué une énorme importance dans le domaine des réseaux sans fil. La technologie des réseaux de capteurs sans fil est devenue une des merveilleuses technologies dans le 21 ème siècle, les réseaux de capteurs ont montré leur impact sur notre vie quotidienne. Le tableau suivant illustre l'évaluation des réseaux de capteurs.

Génération	Période	Taille	Poids	Batterie
1ère	Les années 80 et 90	Grande boite à chaussure	Kilogrammes	Grosse
2ème	Entre 2000 et 2003	Boite de carte	Grammes	AA
3ème	2010	Particule de poussière	Négligeables	Solaire

Tableau I. 1 : Les trois générations des nœuds de capteurs.

3. Réseaux de capteurs sans-fil (RCSF) :

3.1. Qu'est-ce qu'un capteur (senseur) :

C'est un système qui sert à détecter, sous forme de signal souvent électrique, un phénomène physique afin de le représenter. Les capteurs sont des petits appareils dotes d'une batterie, capables de communiquer entre eux et de détecter des événements s'ils se trouvent a l'intérieur de leur rayon de perception. Un capteur est un petit appareil dote de mécanismes lui permettant de relever des informations sur son environnement. La nature de ces informations varie très largement selon l'utilisation qui est faite du capteur : ce dernier peut tout aussi bien faire des relèves de température, d'humidité ou d'intensité lumineuse. Un capteur possède également le matériel nécessaire pour effectuer des communications sans-fil par ondes radio.

3.1.1. Architecture d'un capteur :

Dans cette section, nous distinguons les deux parties qui composent un capteur :

a) Architecture matérielle :

La figure I.1 est l'illustration la plus générale de l'architecture d'un capteur dit intelligent.

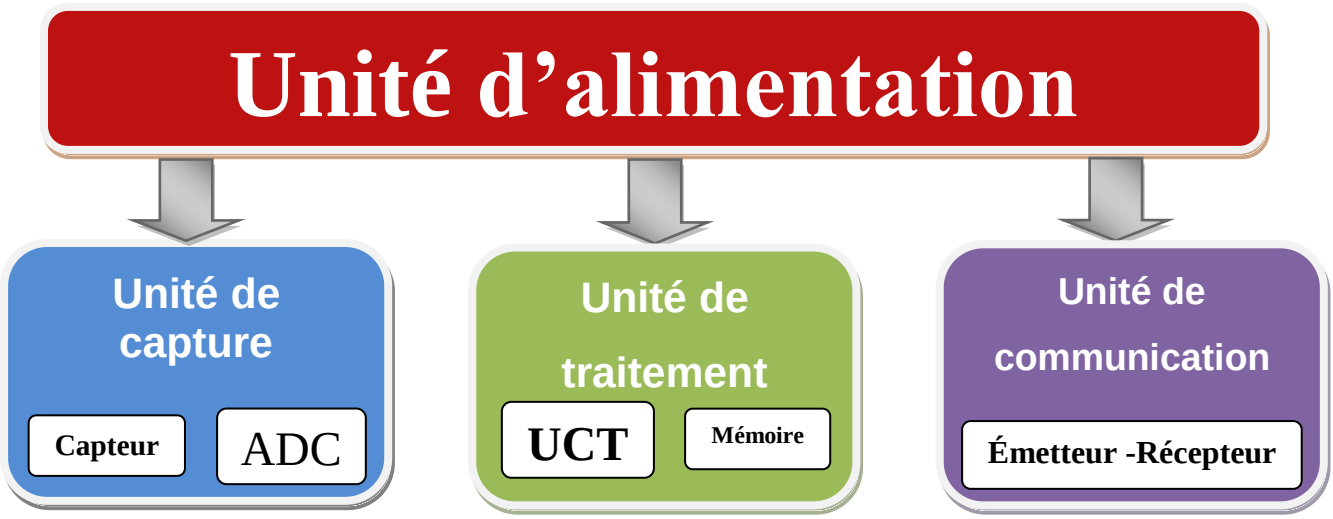

Figure I. 1 : Architecture d'un capteur sans fil.

Cette architecture s'articule autour de quatre unités :

- l'unité de traitement : c'est l'unité principale du capteur. Elle est généralement représentée par un processeur couplé à une mémoire vive. Son rôle est de contrôler le bon fonctionnement des autres unités. Sur certains capteurs elle peut embarquer un système d'exploitation pour

faire fonctionner le capteur. Elle peut aussi être couplée à une unité de stockage, qui servira par exemple à y enregistrer les informations transmises par l'unité d'acquisition de données.

- l'unité d'acquisition : elle permet la mesure des grandeurs physiques ou analogiques et leur conversion en données numériques. Elle est composée du capteur lui-même et de l'ADC1 qui permet la conversion des données. Le capteur est chargé de récupérer les signaux analogiques qu'il transmet à l'ADC qui a pour rôle de transformer et de communiquer les données analogiques en données numériques compréhensibles pour l'unité de traitement.

- l'unité de communication : elle a pour fonction de transmettre et recevoir l'information. Elle est équipée d'un couple émetteur/récepteur pour communiquer au sein du réseau. Il existe cependant d'autres possibilités de transmission (optique, infrarouge, etc..).

- l'unité d'alimentation : c'est un élément primordial de l'architecture du capteur, c'est elle qui fournit en énergie toutes les autres unités. Elle correspond le plus souvent à une batterie ou une pile alimentant le capteur, dont les ressources limitées en font une problématique propre à ce type de réseau puisque ces derniers sont généralement déployés dans des zones non accessibles. La réalisation récente d'unité d'alimentation à base de panneaux solaires tente d'apporter une solution pour prolonger sa durée de vie [8]. Par ailleurs, un capteur peut être doté d'autres unités. Citons, entre autres, la possibilité d'ajouter une unité de localisation, telle qu'un GPS, une unité de mobilité pour assurer la mobilité du capteur, ou une unité spécifique de capture comme une caméra pour de l'acquisition vidéo.

Dans le cas de l'utilisation d'un GPS ou d'une caméra de surveillance, il est intéressant de noter que leur utilisation dans les capteurs actuels a un coût non négligeable en termes de consommation énergétique, qui peut réduire grandement la durée de vie d'un capteur équipé de ce type de dispositifs.

b) Architecture logicielle :

La contrainte énergétique des capteurs exige l'utilisation de systèmes d'exploitation légers tels que TinyOS [9] ou Contiki [10]. Cependant, TinyOS reste toujours le plus utilisé et le plus populaire dans le domaine des RCSF. Il est libre et est utilisé par une large communauté de scientifiques dans des Simulations pour le développement et le test des algorithmes et protocoles réseau.

3.1.2. Types de capteurs :

Il existe actuellement un grand nombre de capteurs, avec des fonctionnalités diverses et variées. La plupart des capteurs dépendent de l'application pour laquelle ils ont été conçus (capteurs aquatiques, sous-terrain, etc.).

Depuis un peu plus de 10 ans, la technologie des capteurs sans fil a beaucoup évolué. Les modules deviennent de plus en plus petits et les durées de vie prévues augmentent.

Aujourd'hui, le marché de nœuds a été ouvert à l'industrie. Le fournisseur le plus connu est Crossbow Inc. avec son offre de capteurs Mica2 et MicaZ.

3.1.3. Caractéristiques principales d'un capteur :

Deux entités sont fondamentales dans le fonctionnement d'un capteur: l'unité d'acquisition qui est le cœur physique permettant la prise de mesure et l'unité de communication qui réalise la transmission de celle-ci vers d'autres dispositifs électroniques. Ainsi, chaque capteur possède un rayon de communication (Rc) et un rayon de sensation (Rs).

La Figure I.2 montre les zones définies par ces deux rayons pour le capteur. La zone de communication est la zone où le capteur peut communiquer avec les autres capteurs. D'autre part, la zone de sensation (ou de détection) est la zone où le capteur peut capter l'événement.

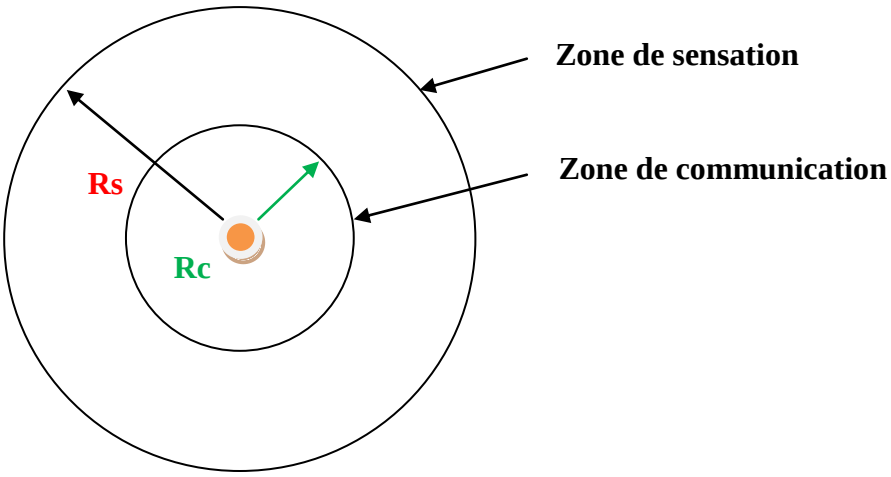

Figure I. 2 : Rayons de communication et de détection d'un capteur.

3.2. Définition d'un RCSF :

Les réseaux de capteurs sans-fil sont considères comme un type spécial des réseaux ad hoc ou l'infrastructure fixe de communication et l'administration centralisée sont absentes et les nœuds jouent, à la fois, le rôle des hôtes et des routeurs. Les nœuds capteurs sont des capteurs intelligents "smart sensors", capables d'accomplir trois tâches complémentaires : la relève d'une grandeur physique, le traitement éventuel de cette information et la communication avec d'autres capteurs. L'ensemble de ces capteurs, déployés pour une application, forme un réseau de capteurs. Le but de celui-ci est de surveiller une zone géographique, et parfois d'agir sur celle-ci (il s'agit alors de réseaux de capteurs-actionneurs). On peut citer comme exemples un réseau détecteur de feu de forêt, ou un réseau de surveillance de solidité d'un

pont après un tremblement de terre. Le réseau peut comporter un grand nombre de nœuds (des milliers). Les capteurs sont placés de manière plus ou moins aléatoire (par exemple par largage depuis un hélicoptère) dans des environnements pouvant être dangereux. Toute intervention humaine après le déploiement des nœuds capteurs est la plupart du temps exclue, le réseau doit donc s'autogérer. Afin que les nœuds capteurs travaillent d'une façon coopérative, les informations recueillies sont partagées entre eux par voie hertzienne. Le choix du lien radio plutôt que du lien filaire permet un déploiement facile et rapide dans un environnement pouvant être inaccessible pour l'être humain.

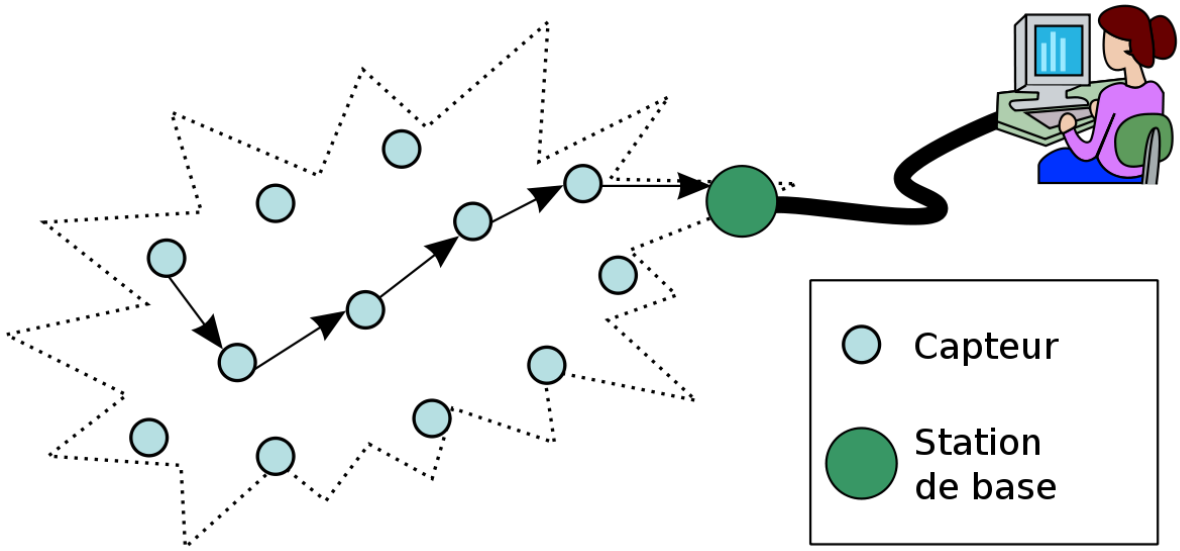

Figure I. 3 : Schéma général d'un réseau de capteurs.

3.2.1. Quelques applications des réseaux de capteurs :

La diminution de taille et de coût des micro-capteurs, l'élargissement de la gamme des types de capteurs disponibles (thermique, optique, vibrations ...) et l'évolution des supports de communication sans fil ont élargi le champ d'application des réseaux de capteurs. Les RCSF peuvent être utilisés dans plusieurs applications [11, 12, 13, 14]. Parmi elles, nous citons :

· **Découverte de catastrophes naturelles :** on peut créer un réseau autonome en dispersant les nœuds dans la nature. Des capteurs peuvent ainsi signaler des événements tels que les feux de forêt, les tempêtes ou les inondations. Ceci permet une intervention beaucoup plus rapide et efficace des secours [15].

· **Détection d'intrusions :** en plaçant à différents points stratégiques des capteurs, on peut ainsi prévenir des cambriolages ou des passages de gibier sur une voie de chemin de fer (par exemple) sans avoir à recourir à de coûteux dispositifs de surveillance vidéo.

· **Gestion de stock :** on pourrait imaginer devoir stocker des denrées nécessitant un certain taux d'humidité et une certaine température. Dans ces applications, le réseau doit pouvoir

collecter ces différentes informations et alerter en temps réel si les seuils critiques sont dépassés.

· **Contrôle de la pollution :** des capteurs au-dessus d'un emplacement industriel offrent la possibilité de détecter et de contrôler des fuites de gaz ou de produits chimiques. Ces applications permettent de donner l'alerte en un temps record et de pouvoir suivre l'évolution de la catastrophe [16].

· **Agriculture :** des nœuds peuvent être incorporés dans la terre et on peut interroger le réseau sur l'état du champ et déterminer par exemple les secteurs les plus secs afin de les arroser en priorité. On peut aussi imaginer équiper des troupeaux de bétail de capteurs pour connaître en tout temps, leur position ce qui éviterait aux éleveurs d'avoir recours à des chiens de berger.

· **Surveillance médicale :** en implantant sous la peau de mini capteurs vidéo, on peut recevoir des images d'une partie du corps en temps réel sans aucune chirurgie. On peut ainsi surveiller la progression d'une maladie ou la reconstruction d'un muscle [17].

· **Surveillance de barrages :** on peut inclure sur les parois des barrages des capteurs qui permettent de calculer en temps réel la pression exercée. Il est donc possible de réguler le niveau d'eau si les limites sont atteintes. On peut aussi imaginer inclure des capteurs entre les sacs de sable formant une digue de fortune. La détection rapide d'infiltration d'eau peut servir à renforcer le barrage en conséquence. Cette technique peut aussi être utilisée pour d'autres constructions telles que ponts, voies de chemin de fer, routes de montagnes, bâtiments et autres ouvrages d'art.

3.2.2. Architecture des réseaux de capteurs :

Puisque les RCSF se caractérisent par l'absence d'une infrastructure déterminée au préalable, les nœuds capteurs la construisent tout en permettant l'interaction avec l'environnement où ils appartiennent et en répondant aux différentes requêtes venant des utilisateurs ou des réseaux externes.

Par ailleurs, les nœuds capteurs comme tout autre composant de télécommunication adhèrent à une architecture protocolaire spécifique. La réalisation de cette dernière requiert la mise en œuvre de techniques développées pour les réseaux Ad Hoc. Cependant, de nouveaux problèmes apparaissent engendrés entre autres par la sévérité des contraintes dues aux limitations de ressources physiques des RCSF. C'est pourquoi il est commode que la conception des protocoles de communication soit faite d'une manière optimale.

a) Architecture de communication :

Après le déploiement des nœuds capteurs sur une certaine zone de captage, ceux-ci commencent par la découverte de leurs voisins afin de construire la topologie de communication. Ainsi, ils deviennent capables d'accomplir les tâches que leur sont affectées. Selon une communication multi-sauts, les capteurs sont chargés de collecter des données, les router vers un nœud particulier appelé nœud puits. Ce dernier analyse ces données et transmet à son tour l'information collectée à l'utilisateur via internet ou bien satellite. Comme l'indique la figure I.4, l'ensemble de nœuds construisant le RCSF est considéré comme étant un réseau d'acquisition de données. Par contre, le réseau de distribution de données est composé des utilisateurs, et du réseau de communication : l'internet, et les satellites.

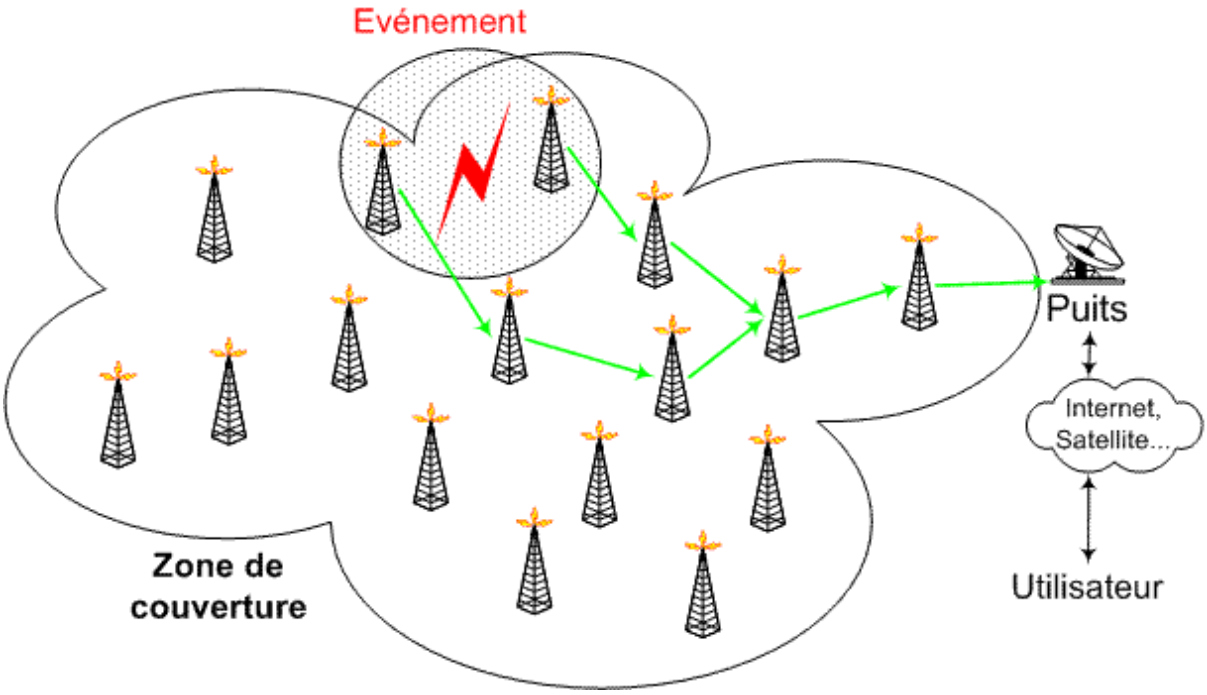

Figure I. 4 : Architecture de communication d'un réseau de capteurs.

b) Architecture protocolaire :

Dans le but d'un établissement efficace d'un RCSF, une architecture en couches est adoptée afin d'améliorer la robustesse du réseau. Une pile protocolaire de cinq couches est donc utilisée par les nœuds du réseau. Citons la couche application, la couche transport, la couche réseau, la couche liaison de données et la couche physique.

De plus, cette pile possède trois plans (niveaux) de gestion : le plan de gestion des tâches qui permet de bien affecter les tâches aux nœuds capteurs, le plan de gestion de mobilité qui permet de garder une image sur la localisation des nœuds pendant la phase de routage, et le plan de gestion de l'énergie qui permet de conserver le maximum d'énergie.

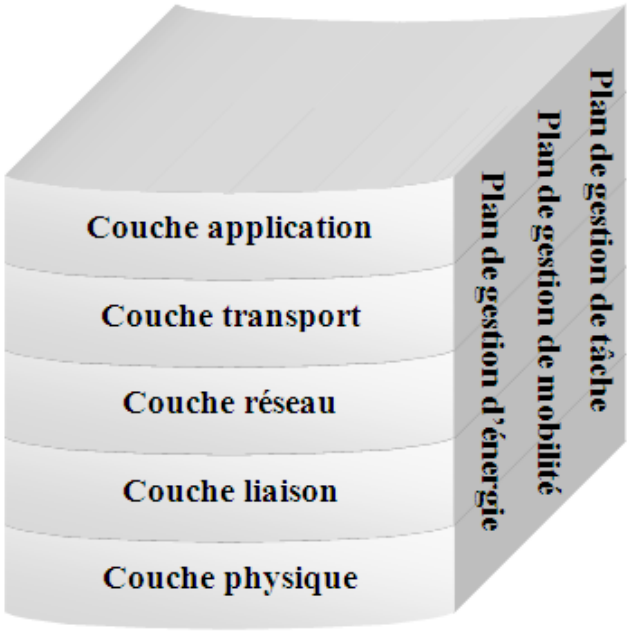

Figure I. 5 : La pile protocolaire dans les réseaux de capteurs [18].

c) Couches de la pile protocolaire [18, 19] :

· **Couche application :** Elle assure l'interface avec les applications. Il s'agit donc de la couche la plus proche des utilisateurs, gérée directement par les logiciels. Parmi les protocoles d'application, nous citons : SMP (Sensor Management Protocol) et TADAP (Task Assignement and Data Advertisement Protocol).

· **Couche transport :** Elle vérifie le bon acheminement des données et la qualité de la transmission. Dans les RCSF, la fiabilité de transmission n'est pas majeure. Ainsi, les erreurs et les pertes sont tolérées. Par conséquent, un protocole de transport proche du protocole UDP et appelé UDP-Like (User Datagram Protocol Like) est utilisé. Cependant, comme le protocole de transport universel est TCP (Transmission Control Protocol), les RCSF doivent donc posséder, lors d'une communication avec un réseau externe, une interface TCP-spliting pour vérifier la compatibilité entre ces deux réseaux communicants.

· **Couche réseau :** Elle s'occupe du routage de données fournies par la couche transport. Elle établit les routes entre les nœuds capteurs et le nœud puits et sélectionne le meilleur chemin en termes d'énergie, délai de transmission, débit, etc. Les protocoles de routage conçus pour les RCSF sont différents de ceux conçus pour les réseaux Ad Hoc puisque les RCSF sont différents selon plusieurs critères comme :

- l'absence d'adressage fixe des nœuds tout en utilisant un adressage basé-attribut.
- l'établissement des communications multi-sauts.

- l'établissement des routes liant plusieurs sources en une seule destination pour agréger des données similaires, etc. Parmi ces protocoles, nous citons : LEACH (Low-Energy Adaptive Clustering Hierarchy) et SAR (Sequential Assignement Routing).

· **Couche liaison de données :** Elle est responsable de l'accès au média physique et la détection et la correction d'erreurs intervenues sur la couche physique. De plus, elle établit une communication saut-par-saut entre les nœuds. C'est-à-dire, elle détermine les liens de communication entre eux dans une distance d'un seul saut.

Parmi les protocoles de liaison de données, nous citons: SMACS (Self-organizing Medium Access Control for Sensor networks) et EAR (Eavesdrop And Register).

· **Couche physique :** Elle permet de moduler les données et les acheminer dans le média physique tout en choisissant les bonnes fréquences.

3.2.3. Les systèmes d'exploitation pour les réseaux de capteurs :

a) TinyOS :

TinyOS est un système d'exploitation open source conçu pour les réseaux de capteurs sans fil. Il est basé sur une architecture orientée composante qui favorise l'implémentation et l'innovation rapide. De plus, il génère un noyau de petite taille (quelques ko), comme l'exigent les contraintes de mémoire imposées par les réseaux de capteurs. [9]

La première implémentation de TinyOS a été réalisée à l'université de Berkeley en 1999 [20]. La version 1.0 est sortie en septembre 2002. La version la plus récente de ce système est la 2.0.2 qui a été disponible en juillet 2007. Ce système d'exploitation est développé avec le langage NesC. Le développement et la maintenance de ce système sont maintenant sous la responsabilité d'un consortium international, TinyOS Alliance.

Figure I. 6 : Logo TinyOS

b) Contiki :

Contiki [21] est un système d'exploitation portable, open source et multitâche pour les réseaux de capteurs. Il supporte beaucoup de plateformes et il a un environnement de simulation netsim.

Ce système a été développé par l'équipe des systèmes embarques dans l'institut des sciences informatique suédoise.

Contiki supporte le multitâche et il implémente la pile protocolaire TCP/IP. Il ne consomme pas beaucoup de mémoire : quelques ko de code et quelques centaines d'octets dans la RAM.

Au contraire de TinyOS qui se base sur la notion d'événements, celui-ci se base sur le multitâche et l'édition statique des liens [22].

4. Conclusion :

Dans ce chapitre, nous avons rappelé certains concepts de base du réseau de capteurs sans fil, du matériel et des logiciels des nœuds de capteurs. En outre, nous avons décrit leurs différentes architectures et caractéristiques. Enfin, les nombreuses applications de la vie réelle de RCSF sont présentées. Dans le chapitre suivant, nous passerons en revue les systèmes de détection d'intrusion, le champ d'application de notre étude.

Chapitre 02

1. Introduction :

La sécurité est un domaine très important pour les RCSF, particulièrement pour des applications sensibles comme le domaine militaire, médical, et autres. La sécurité devrait intervenir pour certaines fonctions sensibles telles que l'expédition des paquets, la gestion du réseau, fonctions effectuées par certains ou tous les nœuds disponibles dans les RCSFs. En raison des différences de base entre les réseaux fixes et des réseaux ad hoc en général, la sécurité dans les RCSF devrait être examinée avec beaucoup plus de minutie.

Dans ce chapitre nous avons mentionné les conditions de la sécurité et les vulnérabilités possibles dans les RCSF. Ensuite, nous décrirons l'IDS et ses moteurs d'analyse qui leur permettent de distinguer les différents comportements.

Nous aborderons en fin du chapitre les techniques de détection d'intrusion et les différentes attaques dans les RCSF.

2. Conditions de Sécurité :

Un réseau de capteur est un type spécial des réseaux Ad Hoc. Il partage quelques vulgarisations avec un réseau informatique typique, mais pose également des conditions uniques de ses propres caractéristiques. Par conséquent, un protocole de sécurité pour un RCSF doit satisfaire une ou plusieurs conditions de sécurité [30], à savoir :

2.1. Confidentialité des Données :

La confidentialité des données est la question la plus importante dans la sécurité de réseau. L'approche standard pour sécuriser le transfert des données est de crypter les données avec une clef secrète connue par l'émetteur et le récepteur

2.2. Intégrité des données :

Un nœud intrus (adversaire) peut modifier les données transférées. Par exemple, un nœud malveillant peut ajouter quelques fragments ou manœuvrer les données dans un paquet.

Ce nouveau paquet peut alors être envoyé au récepteur original. La perte ou les dommages de données peuvent même se produire sans présence d'un nœud malveillant dû à l'environnement de communication. Ainsi, l'intégrité des données s'assure qu'aucune donnée reçue n'a été changée en transit.

2.3. Fraîcheur de Données :

Même si la confidentialité et l'intégrité des données sont assurées, nous devons également assurer la fraîcheur de chaque message. Officieusement, la fraîcheur de données suggère que les données soient récentes, et elles s'assurent qu'aucun vieux message n'a été rejoué. Cette condition est particulièrement importante quand il y a des stratégies de partager clefs utilisées dans la conception. Des clefs typiquement partagées doivent être changées avec le temps.

Cependant, cela prend du temps pour de nouvelles clefs partagées d'être propagées au réseau entier. Dans ce cas-ci, il est facile pour l'adversaire d'employer une attaque de rejouer.

Pour résoudre ce problème, un compteur relatif au temps diffère, peut être ajouté dans le paquet pour assurer la fraîcheur de données.

2.4. Auto-Organisation :

Un réseau de capteur sans fil est typiquement un réseau Ad Hoc, qui exige chaque nœud capteur soit indépendant et assez flexible à l'auto organisation. Il n'y a aucune infrastructure fixe disponible pour la gestion de réseau dans un réseau de capteurs. L'auto organisation apporte un grand défi à la sécurité du réseau de capteurs sans fil [30].

2.5. La Localisation :

Souvent, l'utilité d'un réseau de capteur se fondera sur ses capacités de localiser automatiquement chaque capteur dans le réseau. Un réseau de capteurs conçu pour détecter des anomalies aura besoin de l'information précise d'endroit afin d'indiquer exactement l'endroit d'un défaut

2.6. Authentification :

Un adversaire n'est pas limité simplement à modifier le paquet de données. Il peut changer le jet entier de paquets en injectant les paquets additionnels. Ainsi le récepteur doit s'assurer que les données utilisées dans n'importe quel processus décisionnel proviennent de la source correcte.

D'autre part, en construisant le réseau de capteurs, l'authentification est nécessaire pour beaucoup de tâches administratives (coefficient de reprogrammation ou de contrôle). D'après ce qui précède, nous pouvons voir que l'authentification de message est importante pour beaucoup d'applications dans les réseaux de capteurs. Officieusement, l'authentification de

données permet à un récepteur de vérifier que les données sont vraiment envoyées par l'expéditeur réclamé.

Dans le cas de communication bipartite, l'authentification de données peut être réalisée par un mécanisme purement symétrique : l'expéditeur et le récepteur partagent une clef secrète pour calculer le code d'authentification de message (IMPER) de toutes les données communiquées [30] [31].

3. Vulnérabilités de la sécurité dans les RCSF :

Les principaux problèmes de sécurité dans les RCSF émergent à partir des propriétés qui les rendent efficaces et attrayants, qui sont :

3.1. Limitation de ressources :

L'énergie est peut-être la contrainte la plus forte aux capacités d'un nœud capteur. La réserve d'énergie de chaque nœud doit être conservée pour prolonger sa durée de vie et ainsi que celle de l'ensemble du réseau. Dans la plupart du temps, l'information transmise est redondante vus que les capteurs sont généralement géographiquement Co localisés.

La plupart de cette énergie peut donc être économisée par agrégation de données. Cela exige une attention particulière à détecter l'injection de fausses données ou la modification défectueuse de données, lors des opérations d'agrégation au niveau des nœuds intermédiaires.

3.2. La communication sans fils multi-sauts :

En plus de fournir un déploiement simple, la communication sans fil a l'avantage d'offrir l'accès à des endroits difficilement accessibles tels que des terrains désastreux et hostiles. Malheureusement, la portée de la communication radio des "Motes" est limitée en raison de considérations énergétiques. La communication multi-sauts est donc indispensable pour la diffusion des données dans un RCSF. Cela introduit de nombreuses failles de sécurité à deux niveaux différents : attaque de la construction et maintenance des routes, et attaque des données utiles par injection, la modification ou la suppression de paquets. En outre, la communication sans fil introduit d'autres vulnérabilités à la couche liaison en ouvrant la porte à des attaques de brouillage et de style déni de service par épuisement des batteries.

3.3. Couplage étroit avec l'environnement :

La plupart des applications de RCSF exigent un déploiement étroit des nœuds à l'intérieur ou à proximité des phénomènes à surveiller. Cette proximité physique avec l'environnement

conduit à de fréquentes compromissions intentionnelles ou accidentelles des nœuds. Comme le succès des applications RCSF dépend également de leur faible coût, les nœuds ne peuvent pas se permettre une protection physique inviolable.

Par conséquent, un adversaire "bien équipé" peut extraire des informations cryptographiques des nœuds capteurs. Comme la mission d'un RCSF est généralement sans surveillance, le potentiel d'attaquer les nœuds et de récupérer leur contenu est important.

Ainsi, les clefs cryptographiques et informations sensibles devraient être gérées d'une manière qui augmente la résistance à la capture des nœuds

3.4. Mécanisme de sécurité :

Pour contrer les attaques qui menacent les réseaux de capteurs sans fil, plusieurs équipes de recherche tentent de trouver des solutions appropriées. Ces solutions doivent bien sûr prendre en compte les spécificités des réseaux de capteurs sans fil. Il faut donc trouver des solutions simples qui permettent de sécuriser le réseau tout en consommant le moins d'énergie possible et adapter ces solutions à une puissance de calcul faible.

Dans l'éventail de ces solutions, on trouve des mécanismes tels que le partitionnement de données, l'utilisation de méthodes cryptographiques adaptées, la détection d'intrus par localisation ou bien encore l'indice de confiance.

4. Détection d'intrusion :

La détection d'intrusion est l'acte de détecter les actions qui essaient de compromettre la confidentialité, l'intégrité ou la disponibilité d'une ressource. La détection d'intrusion peut être effectuée manuellement ou automatiquement. Dans le processus de détection d'intrusion manuelle, un analyste humain procède à l'examen de fichiers de logs à la recherche de tout signe suspect pouvant indiquer une intrusion.
Un système qui effectue une détection d'intrusion automatisée est appelé système de détection d'intrusion (IDS).

5. Les différents types d'intrusion :

5.1. Le cheval de Troie :

Programme malveillant d'apparence anodine (jeux, utilitaire) qui, une fois installé, peut causer des dégâts ou permettre la prise de contrôle à distance de l'ordinateur infecté, et ainsi fournir au pirate, une porte ouverte au contenu du système d'information de l'entreprise

5.2. Le ver (Worm) :

Programme capable de fonctionner de manière indépendante. Il se propage de machine en machine. Un ver ne modifie aucun programme, mais il peut transporter des séries de codes informatiques qui pourront le faire (des virus par exemple).

5.3. Le virus :

Capable d'infecter d'autres programmes en les modifiant pour y inclure une copie de lui-même. Le virus nécessite l'exécution du programme hôte pour s'activer. Il se multiplie au sein de l'environnement qu'il cible et entraine corruption, perturbation et/ou destruction.

5.4. Les bombes logiques :

Programmes destructeurs à déclenchement différé. Par exemple, un programmeur insère dans le programme de paie de l'entreprise qui l'emploie une fonction de destruction dont l'exécution sera déclenchée si le nom du programmeur disparaît du fichier du personnel.

5.5. L'attaque en déni de service :

Activité consistant à empêcher quelqu'un d'utiliser un service par saturation d'exécution de programmes.

5.6. Le pourriel ou spam :

Communication électronique, non sollicitée par les destinataires et expédiée en masse à des fins publicitaires ou malhonnêtes.

5.7. L'adware :

Permet d'afficher des bannières publicitaires

5.8. Le spyware :

Installe sur le poste de l'utilisateur un logiciel espion et envoient régulièrement et sans accord préalable de l'utilisateur, des informations statistiques sur les habitudes de celui-ci.

5.9. L'hameçonnage :

Consiste à adresser à une personne un courriel de sa banque par exemple, lui expliquant qu'à la suite d'un problème informatique ou d'un changement de logiciel, cette personne doit confirmer ses codes d'accès pour pouvoir continuer à consulter ses comptes, faire des transferts de fonds, etc. Dans ce message figure un lien permettant d'accéder à la page de confirmation des codes en question. Évidemment cette page web reproduit fidèlement la

charte graphique du site officiel de l'institution bancaire, ce qui ne manquera pas de rassurer l'utilisateur alors convaincu de dialoguer avec sa banque.

6. Définition d'un IDS (système détection d'intrusion) :

On appelle IDS (Intrusion Détection System) un mécanisme écoutant le trafic réseau de manière furtive afin de repérer des activités anormales ou suspectes et permettant ainsi d'avoir une action de prévention sur les risques d'intrusion.

Il existe deux grandes familles distinctes d'IDS :

- Les N-IDS (Network Based Intrusion Détection System), ils assurent la sécurité au niveau du réseau.
- Les H-IDS (Host Based Intrusion Détection System), ils assurent la sécurité au niveau des hôtes.

Un N-IDS nécessite un matériel dédié et constitue un système capable de contrôler les paquets circulant sur un ou plusieurs liens réseau dans le but de découvrir si un acte malveillant ou anormal a lieu. Le N-IDS place une ou plusieurs cartes d'interface réseau du système dédié en mode promiscuité (promiscuous mode), elles sont alors en mode « furtif » afin qu'elles n'aient pas d'adresse IP. Elles n'ont pas non plus de pile de protocole attachée. Il est fréquent de trouver plusieurs IDS sur les différentes parties du réseau et en particulier de placer une sonde à l'extérieur du réseau afin d'étudier les tentatives d'attaques ainsi qu'une sonde en interne pour analyser les requêtes ayant traversé le pare-feu ou bien mener depuis l'intérieur.

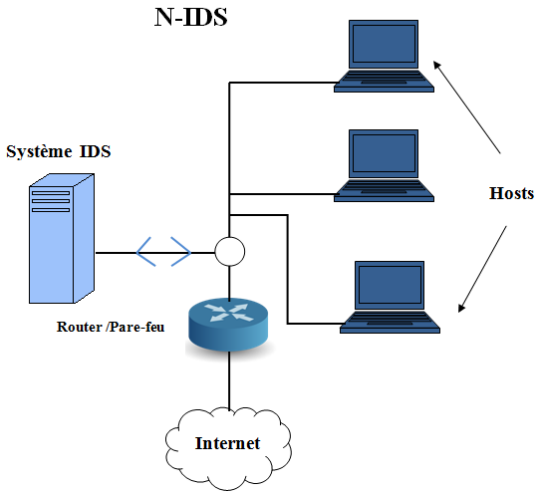

Figure II. 1 : Les N-IDS

Le H-IDS réside sur un hôte particulier et la gamme de ces logiciels couvre donc une grande partie des systèmes d'exploitation tels que Windows, Solaris, Linux, etc. Le H-IDS se comporte comme un démon ou un service standard sur un système hôte. Traditionnellement, le H-IDS analyse des informations particulières dans les journaux de logs (syslogs, messages, lastlog …) et aussi capture les paquets réseau entrant/sortant de l'hôte pour y déceler des signaux d'intrusions (Déni de Services, chevaux de Troie, tentatives d'accès non autorisés, exécution de codes malicieux…).

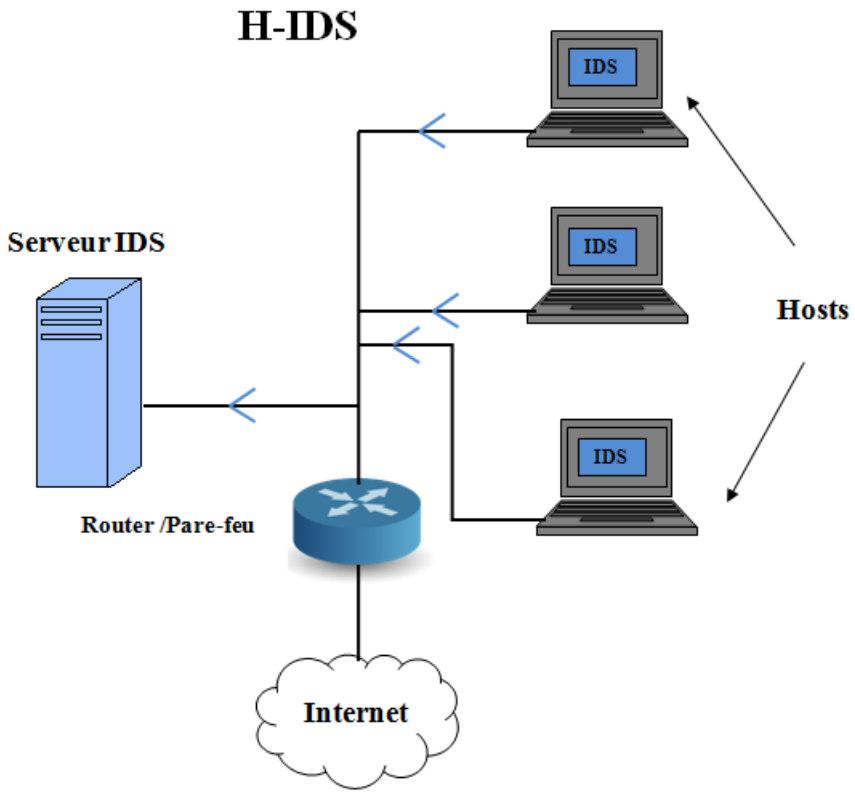

Figure II. 2 : Les H-IDS

7. Les techniques de détection d'intrusion :

7.1. La détection d'abus (misuse détection) :

Aussi appelée détection de mauvaise utilisation, l'IDS analyse l'information recueillie et la compare avec une base de données de signatures d'attaques connues toute activité correspondante est considérée comme une attaque.

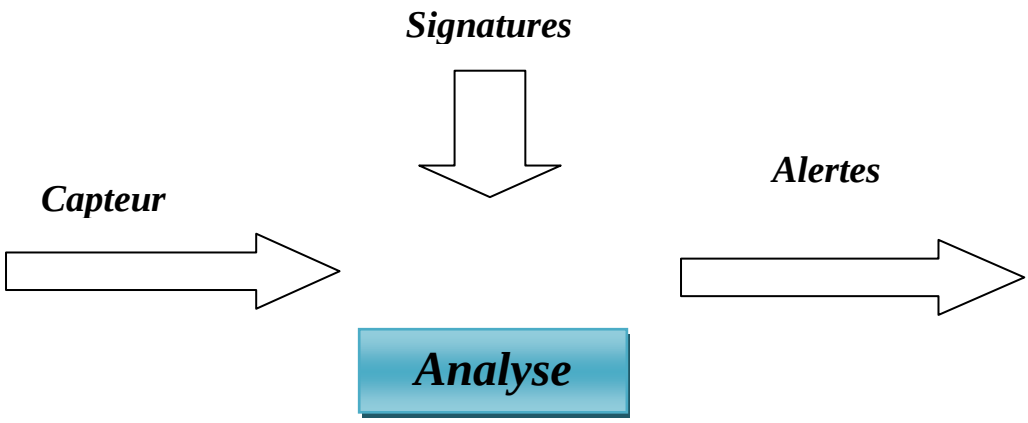

Figure II. 3 : la méthode de détection par signatures

Avantage

1. Simplicité de mise en œuvre.
2. Rapidité de diagnostic.
3. Précision (en fonction des règles).
4. Identification du procédé d'attaque.

Inconvénients

1. Ne détecte que les attaques connues.
2. Maintenance de la base.

7.2. La détection d'anomalie (anomaly detection) :

La détection d'anomalie de comportement est une technique assez ancienne elle est utilisée également pour détecter des comportements suspects en téléphonie, cette technique basée sur le comportement « normal » du système

- ❖ Une déviation par rapport à ce comportement est considérée suspecte.
- ❖ Le comportement doit être modélisé on définit alors un profil.
- ❖ Une attaque peut être détectée sans être préalablement connue.

Avantages

1. Permets la détection d'attaque inconnue.
2. Facilite la création de règles adaptées à ces attaques.
3. Difficile à tromper.

Inconvénients

1. Les faux-positifs sont nombreux
2. Générer un profil est complexe
3. Diagnostics long et précis en cas d'alerte

8. Présentation des attaques :

Les différentes spécificités des réseaux de capteurs sans fil (énergie limitée, faible puissance de calcul, utilisation des ondes radio, etc..) Les exposent à de nombreuses menaces.

Si certaines de ces menaces peuvent se retrouver dans les réseaux ad-hoc d'autres sont spécifiques aux réseaux de capteurs sans fil et s'attaquent plus particulièrement à l'énergie limitée des capteurs.

On parlera d'attaque active si un attaquant modifie l'état du réseau, et d'attaque passive dans le cas où il ne cherchera qu'à l'écouter.

La plupart des attaques de la couche réseau contre les réseaux de capteurs appartiennent à l'une des catégories suivantes :

8.1. Destruction ou vol :

Les plus élémentaires des attaques actives dans les réseaux de capteurs sans fil sont le vol ou la destruction des capteurs. Les capteurs sont déployés dans des zones qui ne peuvent être toujours surveillées. Ainsi une personne physique seule peut subtiliser un ou plusieurs capteurs, voire peut les détruire. Si un capteur est détruit, le réseau doit être capable de s'adapter à la nouvelle situation et éviter d'être divisé en plusieurs sous-réseaux incapables de communiquer entre eux.

De plus, un nœud volé peut divulguer certaines informations à un attaquant. Il peut tout aussi bien être reprogrammé et être réinséré dans le réseau et ainsi devenir un nœud malicieux, fonctionnant en tant que nœud-espion comme expliqué dans [23] [24] et [25].

8.2. Attaque spécifique au type de capteur :

Ce type d'attaque dépend du type de capteur utilisé sur le réseau.

Un attaquant va modifier de manière physique le comportement du capteur. Il peut par exemple allumer une flamme devant un capteur thermique ou bien allumer une lampe devant un capteur de luminosité. Le but est de tromper le capteur, et ainsi d'envoyer ou d'enregistrer de fausses informations sur le réseau, ou bien tout simplement de faire réagir assez longtemps un nœud ou le réseau pour qu'ils consomment leur énergie.

8.3. L'écoute passive :

Cette attaque consiste à écouter le réseau et à intercepter les informations circulant sur le médium. Cette attaque est facilement réalisable si les messages circulant sur le réseau sont en clair. Par ailleurs cette attaque est difficile à détecter, car comme elle est passive, elle ne modifie pas l'activité du réseau.

8.4. Brouillage radio :

Un attaquant va envoyer des ondes sur la même fréquence que le réseau de capteurs sans fil [30]. Ainsi les nœuds ne pourront plus communiquer, car le médium est saturé par le brouillage radio.

8.5. L'injection de messages :

L'attaquant va chercher par divers moyens à injecter des messages dans le réseau. Le but peut être de faire circuler de fausses informations ou tout simplement de saturer le réseau.

8.6. Flooding :

Un attaquant va utiliser un ou plusieurs nœuds malicieux ou un dispositif particulier avec une puissance d'émission forte, pour envoyer régulièrement des messages sur le réseau pour le saturer.

On est en présence d'une attaque active qui est de même type que les attaques de type déni de service dans les réseaux classiques [30].

8.7. Hello Flooding :

Les protocoles de découvertes sur les réseaux ad-hoc utilisent ce qu'on appelle des messages de type HELLO pour s'insérer dans un réseau et pour découvrir ses nœuds voisins.

Dans une attaque dite de HELLO Flooding, un attaquant va utiliser ce mécanisme pour saturer le réseau et consommer son énergie.

Dans [26], on trouve un exemple, représenté par la figure II.4, d'un nœud malicieux X avec une connexion puissante qui lui permet d'envoyer à un grand nombre de nœuds des messages de type HELLO, de manière continue. Les nœuds voisins V vont alors essayer de lui répondre, même s'ils sont situés à des distances qui ne permettent pas d'atteindre le nœud malicieux. À force de tenter de répondre à ces messages, ils vont petit à petit consommer l'intégralité de leur énergie.

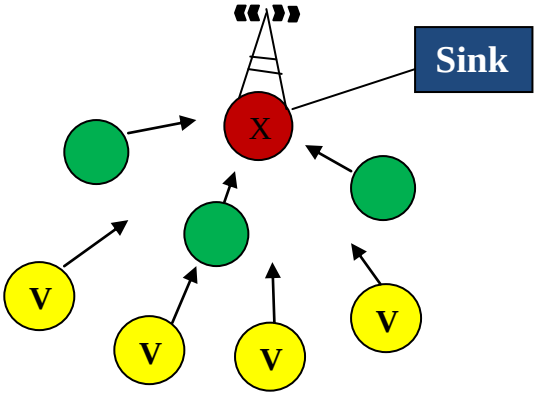

Figure II. 4 : Attaque de type HELLO Flooding

8.8. La privation de mise en veille :

Cette attaque active a pour but de priver un capteur de se mettre en veille par différents moyens [27]. Le capteur s'il ne peut plus se mettre en veille va consommer très rapidement sa batterie, jusqu'à se retrouver hors service.

8.9. Insertions de boucles infinies :

Un attaquant va modifier le routage du réseau avec un ou plusieurs nœuds malicieux, dans le but d'envoyer des messages qui vont être routés en boucles infinies et vont donc consommer l'énergie du réseau.

8.10 L'altération de message :

Un nœud malicieux va récupérer un message et l'altérer, en lui ajoutant des fausses informations (sur le destinataire, l'émetteur ou les données), en le modifiant ou bien en détruisant des paquets pour rendre incompréhensible le message.

8.11 Ralentissement :

Un attaquant peut programmer des nœuds malicieux qui seront comme des agents dormants et qui n'auront que pour but de ralentir l'information (par exemple avec une attaque de type trou gris).

8.12 Attaque du trou noir (Blackhole attack) :

L'attaque du trou noir consiste tout d'abord à insérer un nœud malicieux dans le réseau [26]. Ce nœud, par divers moyens, va modifier les tables de routage pour obliger le maximum de nœuds voisins à faire passer l'information par lui. Ensuite comme un trou noir dans l'espace, toutes les informations qui vont passé en son sein ne seront jamais retransmises.

La figure II.5 représente un trou noir mis en place par un nœud malicieux X qui a modifié le routage pour que les clusters 1, 2, 3 et 4 fassent passer l'information par lui pour communiquer entre clusters. Dans ce cas de figure, le trou noir X ne retransmettra aucune information, empêchant toute communication entre les différents clusters.

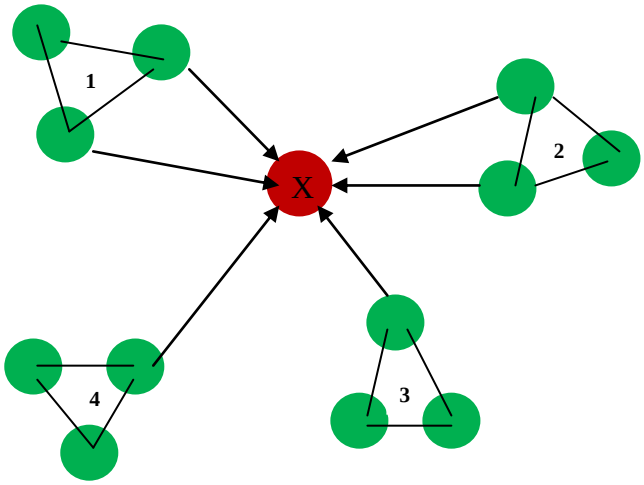

Figure II. 5 : Exemple de trou noir dans un réseau clustérisé.

8.13 Attaque du trou gris (Greyhole attack) :

L'attaque du trou gris est une variante améliorée de l'attaque du trou noir [26]. Contrairement au trou noir, le trou gris relaye certaines informations. Par exemple, le trou gris va relayer toutes les informations concernant le routage, sauf pour des informations critiques. Ce type d'attaque est ainsi plus difficile à détecter que l'attaque du trou noir, le capteur malicieux tant qu'il se comporte de manière normale ne peut être détecté.

8.14 Sybil attack :

Une attaque de type "Sybil attack" [28] consiste à ce qu'un capteur malicieux se fasse passer pour plusieurs capteurs. Il va ainsi pouvoir modifier la table de routage qui deviendra caduque. Un nœud malicieux qui peut se faire passer pour plusieurs nœuds peut gagner un avantage important pour une élection de nœud maître par exemple.

8.15 L'attaque du trou de ver (wormhole attack) :

L'attaque du trou de ver nécessite l'insertion d'au moins deux nœuds malicieux [29]. Ces deux nœuds sont reliés entre eux par une connexion puissante par exemple une liaison filaire.
Le but de cette attaque est de tromper les nœuds voisins sur les distances. Généralement le protocole de routage cherche le chemin le plus court en nombre de sauts (hop).

Dans le cas d'une attaque du trou de ver, les deux nœuds malicieux permettent d'atteindre un lieu éloigné avec un saut unique. Cette possibilité va tromper les autres nœuds sur les distances réelles qui séparent les deux nœuds, mais va surtout obliger les nœuds voisins à passer par les nœuds malicieux pour faire circuler les informations. Ainsi les nœuds malicieux qui forment le trou de ver vont se trouver dans une position privilégiée qui va leur permettre d'avoir une priorité sur l'information circulant à travers leurs nœuds proches.

Cette attaque est représentée par la figure II.6 où deux nœuds malicieux X1 et X2, reliés par une connexion puissante, forment un trou de ver. Les nœuds A et B vont alors privilégier la route la plus rapide formée par le trou de ver, et donc l'information pourra être récupérée par l'attaquant.

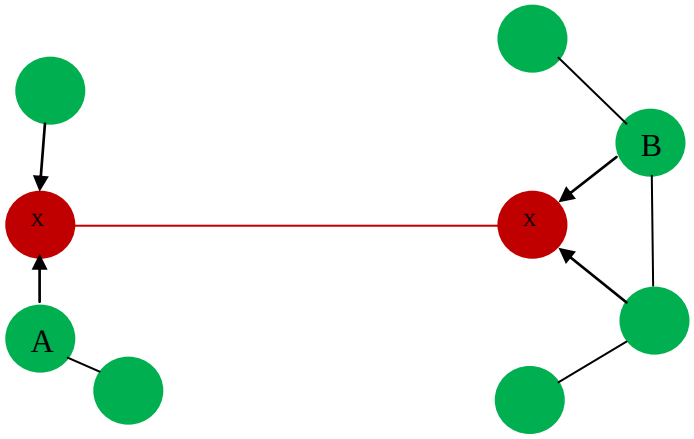

Figure II. 6 : Exemple d'une attaque de type trou de ver.

8.16 L'attaque du trou de la base (sinkhole attack) :

Dans cette attaque un nœud malicieux va s'attaquer directement à l'information circulant par la base, qui est le plus souvent le point qui recueille le plus d'informations de l'intégralité du réseau [26]. Pour cela, le nœud malicieux va proposer aux nœuds le chemin le plus rapide pour atteindre la base, en utilisant une connexion plus puissante.

Ainsi l'ensemble de ces nœuds va s'adresser en particulier à ce nœud malicieux pour transmettre l'information à la base. Toutes les informations qui transitent de ces nœuds vers la base pourront être récupérées par l'attaquant.

Pour générer une attaque encore plus puissante, un attaquant peut utiliser des stratégies de type trou de ver associées à une attaque de type trou de la base. Le but sera avec ces trous de ver de couvrir tous les nœuds du réseau. Cette situation est représentée dans la figure II.7, où les nœuds malicieux X1, X2 et X3 sont reliés par des connexions puissantes et forment des trous de ver. X3 est lui relié à la base par une connexion puissante pour réaliser une attaque

du trou de la base. On parle alors d'une sphère d'influence exercée par l'attaquant sur le réseau, car il est ainsi capable de récupérer l'intégralité des informations qui circulent dans le réseau de capteurs sans fil.

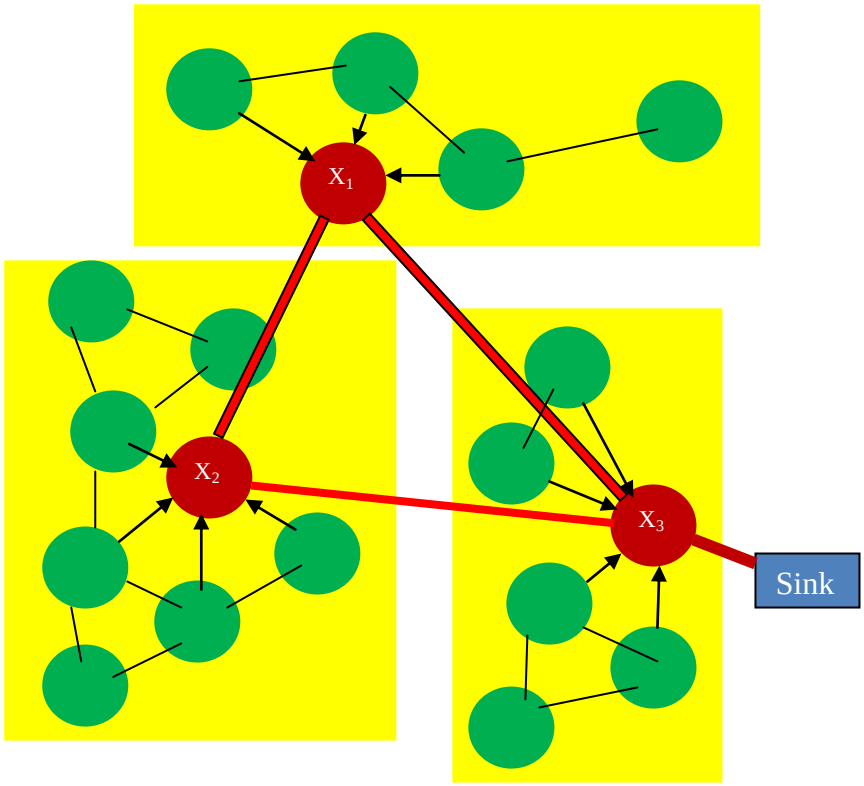

Figure II. 7 : Exemple d'utilisation d'attaques de type trou de ver pour réaliser une attaque de type trou de la base

9 Conclusion :

Dans ce chapitre, nous avons donné un aperçu de la sécurité dans les réseaux de capteurs sans fil et nous avons parlé du système de détection d'intrusion.

Dans le prochain chapitre, nous discuterons de notre approche, des expériences et des résultats.

Chapitre 03

1. Introduction :

La menace que représente la recrudescence des attaques par déni de service (DoS) reste une préoccupation majeure pour de nombreux acteurs de réseau de capteur sans fil.

Avant de pouvoir être neutralisées, ces attaques doivent être identifiées l'aide de détecteurs capables de traiter le trafic en temps réel, aussi autonomes que possible et devant fournir les informations nécessaires à une décision appropriée. Bien que les méthodes statistiques récentes aient permis la création de détecteurs plus autonomes, les nombreux faux positifs qu'ils produisent et les ressources de calcul conséquentes qu'ils nécessitent ont rendu ces détecteurs peu compétitifs. Ainsi, malgré l'engouement durable de la communauté scientifique pour ce sujet, aucun détecteur ne semble faire véritablement consensus. Pour répondre à cette problématique, nous proposons un système de détection d'intrusion IDS basé sur l'analyse de la variance (ANOVA / MANOVA),

Dans ce chapitre, nous détaillons notre système IDS et pour faire, nous allons concevoir notre jeu de donnée ou les attaques DoS seront caractérisées par un ensemble d'attributs. Nous justifierons tous les choix effectués en ce qui concerne le niveau de caractérisation des attaques et les caractéristiques ciblées. Ensuite, nous décrivons notre étude expérimentale et ses résultats donnés.

2. Schémas de détection proposé :

La méthode de détection est basée sur l'anova, avant la détailler on va expliquer le fonctionnement de ce test.

2.1 Analyse de la variance (ANOVA) :

L'analyse de la variance (ou test ANOVA en anglais ANalysis Of VAriance) est un test statistique qui permet de comparer globalement l'espérance mathématique de plusieurs échantillons. Le nom de ce test s'explique par sa façon de procéder : on décompose la variance totale de l'échantillon en deux variances partielles, la variance Inter-classes et la variance résiduelle, et on compare ces deux variances.

Données : p groupes d'observations, avec pour chaque groupe k des observations $(x_{k,1},...,x_{k,nk})$ d'une variable aléatoire X_k d'espérance mathématique μ_k . On note $N=n_1+...+n_p$ le nombre total de valeurs observées.

- Les variances des échantillons sont homogènes (hypothèse d'homoscédasticité).

L'ANOVA est très sensible à cette condition. Le test de Levene permet de tester si les variances entre les échantillons sont égales. Les hypothèses du test de Levene sont les suivantes :

* H_0 : les variances entre les échantillons sont égales,
* H_1 : les variances entre les échantillons ne sont pas égales.

- **Déroulement du test :**

1. On calcule m_k la moyenne empirique de chaque classe : $m_k = \frac{m_{k,1} + \ldots + x_{k,n_k}}{n_k}$
2. On calcule M la moyenne empirique totale de l'échantillon :

$$M = \frac{n_1 m_1 + \cdots + n_p m_p}{N}$$

3. On calcule la variance empirique V_k de chaque classe :

$$V_k = \frac{1}{n_k} \sum_{i=1}^{n_k} (x_{k,i} - m_k)^2.$$

4. On calcule la moyenne des variances, ou variance intra-classes :

$$V_{intra} = \sum_{k=1}^{p} \frac{n_k}{N} V_k.$$

5. On calcule la variance des moyennes, ou variance Inter-classes :

$$V_{inter} = \sum_{k=1}^{p} \frac{n_k}{N} (m_k - M)^2.$$

6. On calcule la variable de test

$$F_{p-1,N-p} = \frac{V_{inter}/(p-1)}{V_{intra}/(N-k)}.$$

7. On compare avec la valeur critique de la loi de Fischer-Snedecor de degrés de liberté q-1 et N-q pour le risque a voulu. Si la variable de test est supérieure à

la valeur critique, alors on rejette l'hypothèse. Ou bien Si p-value < 0,05, l'hypothèse nulle est rejetée, cela signifiant qu'il y a une différence significative entre les variances des échantillons. L'homoscédasticité est donc supposée si p-value > 0,05.

- **Limitations :** En théorie, les variables aléatoires X_k doivent être des variables aléatoires de même variance. Toutefois, ce test est robuste, si les échantillons sont grands et les variances pas trop différentes, le résultat est tout de même significatif.

L'analyse de la variance est notamment utilisée lorsqu'on souhaite comparer l'efficacité de plusieurs traitements. Par exemple, si on a plusieurs groupes de personnes à qui on donne un médicament contre le cholestérol, on souhaite savoir si ces médicaments sont efficaces et font baisser significativement le taux de cholestérol. On parle alors d'analyse de la variance à un facteur : on teste l'efficacité d'un facteur sur la variable. On peut généraliser ces méthodes aux cas de plusieurs facteurs (par exemple la combinaison de plusieurs médicaments...).

- Analyse de variance avec R :

L'ANOVA permet de voir si une variable numérique a des valeurs différentes en fonction de plusieurs groupes. C'est une généralisation du test de Student permettant de comparer plus de deux groupes.

La commande à utiliser est :

Code R :

```
Fit <- lm(y ~ A, data= mydataframe) # y est la variable numérique et A indique les groupes
anova(Fit)
```

Exemple :

Hypothèse testée : "Les espérances sont égales".

Code R :

```
fr<- data.frame (var = c(10, 4, 5, 3, 3, 7, 2, 6, 2, 8, 5),
group = factor(c ("c", "a", "b", "a", "b", "b", "a", "b", "a", "c", "c"))) # table de données
Fit <- lm (var ~ group, fr)# analyse de variance
anova(Fit)
```

Résultat de la commande:

```
            Df Sum Sq Mean Sq F value  Pr(>F)
group        2 41.833 20.9167  6.9241 0.01798 *
Residuals    8 24.167  3.0208
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Le test renvoie une p-value significative; p = 0.018.

- Nous remarquons que la variable de test (p) est inférieure à la valeur critique 0.05, alors on rejette l'hypothèse.

2.2. Méthode de détection :

L'attaque DoS vise à perturber le fonctionnement normal et les informations (attributs) du réseau. Avant de mettre en place notre méthode de détection, il nous a fallu déterminer quels sont les attributs les plus sensibles aux attaques DoS, pour les utiliser comme des indices révélateurs d'attaques DoS. Ensuite le Sink, peut effectuer le test afin de déterminer la présence d'une attaque. Notre algorithme suit les étapes suivantes :

- Le sink collecte les valeurs des attributs choisis au préalable pour une période de temps fixe, ces valeurs seront la référence d'un fonctionnement normal du réseau.
- Après, le sink collecte à nouveau des nouvelles valeurs qui seront comparais en utilisant l'ANOVA avec les valeurs références.
- Si le test échoue, on considère qu'il n'y a pas d'attaque sur le réseau et le sink passe à des nouvelles valeurs.
- Sinon, le sink alerte le réseau sur la présence d'une attaque DoS.

Dans la partie suivante nous détaillons comment nous avons implémenté notre approche, les outils qu'on a utilisés ainsi que les expérimentations qu'on a effectuées.

3. Simulations et Expérimentations :

Pour tester notre approche, nous avons fait appel à la simulation pour générer des données de tests. Pour vérifier l'efficacité de notre approche. Dans la suite, nous détaillons comment nous avons généré ces données à partir de la simulation.

3.1 Génération des données :

Pour générer des données, nous avons utilisé le simulateur NS-2 ainsi que d'autres outils comme GAWK, Rstudio.

3.1.1 Vue globale :

On a suivez plusieurs étapes afin de construire un système IDS, ce travail est illustré dans la figure III.1.

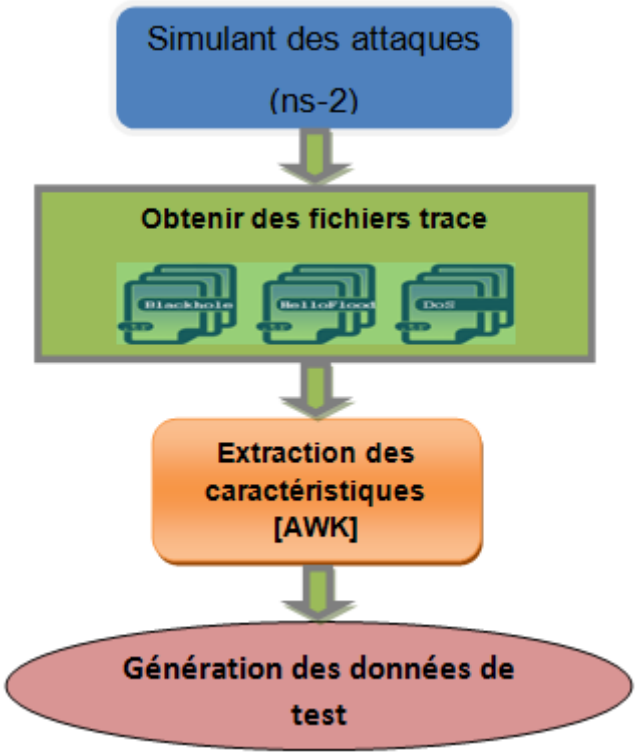

Figure III. 1 : Vue globale du système.

3.1.2 Simulation avec ns2 :

3.1.2.1 Modèle d'application :

Figure III. 2 : Modèle d'application.

Nous avons considéré un environnement simple, où les nœuds de capteurs sont statiques et placés dans une topologie maillée comme le montre la Figure III.2. Le nœud avec

l'identification "0" est considéré comme le Sink. Nous avons utilisé 70 nœuds dans nos simulations. Ce choix est fait pour faciliter la visualisation.

Rappelons que le modèle de diffusion de données est piloté par les événements. Les nœuds détectent des valeurs telles que la température, la pression, le son, etc. Lorsque ces valeurs atteignent un certain seuil, elles sont transmises au Sink.

Concernant le routage, nous avons effectué notre analyse en utilisant AODV. Notre choix est justifié par son adéquation avec RCSF. En plus de la capacité à soutenir la mobilité.

Plus de détails sur les configurations ns-2 sont présentés à l'annexe A de la section 1 à la section 4.

3.1.2.2 Les attaques simulées :

Nous avons simulé le comportement normal et trois comportements malveillants différents: Blackhole, Hello Flood et DoS.

Concernant le comportement Normal, nous avons choisi le protocole AODV pour le routage et le protocole TCP pour la transmission. Il y a 70 nœuds dans notre topologie, alors que le plan de livraison est à un moment donné, un nœud envoie des données au sink (nœud 0). Le nœud commence à envoyer des données au hasard, ce qui nous permet de simuler des applications réelles. Le script TCL ns-2 du comportement normal est présenté à l'annexe A, section 8.

Pour reproduire le même scénario Normal lors de la simulation d'attaques, nous avons sélectionné le scénario (Sauvegarde de l'identité des nœuds de travail et de leur temps de début de transmission de données). Le script TCL ns-2 à exécuter est détaillé à l'annexe A, section 5.

En ce qui concerne les comportements malveillants, nous avons généré la même topologie que le comportement normal avec les mêmes paramètres, puis nous chargeons le fichier qui contient les identités des nœuds et leur temps de départ des données d'envoi. Ensuite, nous avons choisi au hasard un seul nœud pour qu'il agisse de manière malveillante. Nous avons choisi trois comportements malveillants différents:

- **Blackhole** : Le nœud malveillant supprime tous les paquets qui le traversent. Pour ce faire, il attire ses nœuds voisins en forgeant une réponse d'itinéraire avec moins de nombres de sauts et un plus grand nombre de séquences. Nous avons simulé cela en

ajoutant cet acte malveillant au protocole AODV, comme indiqué à l'annexe A, section 9.1, et au script TCL, section 9.2.

- **Hello Flood :** Le nœud flooder continue d'envoyer des messages RREQ malgré la réception de messages RREP, dans le but de gaspiller la bande passante du réseau et d'épuiser ses ressources. Nous avons simulé ceci en ajoutant le code indiqué à l'annexe A, section 10.1, au protocole AODV et le script TCL est détaillé à la section 10.2.
- **DoS :** Le nœud attaquant continue d'envoyer des paquets au récepteur, dans le but de le rendre inactif. Le script TCL est présenté à l'annexe A, section 11.

En suivant les scénarios décrits précédemment, nous avons lancé les simulations. Cela a conduit à un ensemble de fichiers de trace à partir desquels nous avons extrait les attributs ciblés. Nous avons déployé des scripts AWK pour effectuer cette extraction. Le script est reporté en annexe B. Chaque connexion est caractérisée par un ensemble d'attributs.

3.1.2.3 Les paramètres de simulations :

Pour prouver l'efficacité de notre solution dans la détection de l'attaque DoS et évaluer sa performance, nous avons effectué une série de simulation en utilisant les paramètres de simulation listés dans le tableau 4.1 :

Paramètre	Signification
Le nombre de noeuds	70
Taille du réseau	200*200 m²
portée de communication	25m
Taille du paquet	1024 bits
Protocole de routage	AODV
Temps de simulation	80s

Tableau III. 1 : Paramètres de simulation.

3.1.2.4 Les attributs collectés :

De nombreux attributs peuvent être considérés pour caractériser un comportement au niveau de la connexion. Dans notre étude, nous avons examiné certains d'entre eux comme indiqué dans le tableau 3.1. Nous avons classé ces caractéristiques en fonction de leur type: Données, Routage et Temps.

Attributs	Types	La description
Durée	Temps	La durée entre l'envoi du premier paquet et la réception du dernier paquet.
Toutes les données de paquets reçus	Données, routage	Tous les paquets (indépendamment de qui les a envoyés et de leur type) que le récepteur a reçus pendant la durée de cette connexion
Demande de route AODV	Routage	Nombre de paquets AODV RREQ envoyés par le nœud expéditeur.
AODV Route Réponse	Routage	Le nombre de paquets RREP AODV envoyés par le nœud émetteur.
Paquet TCP envoyé	Données	Le nombre de paquets TCP envoyés par le noeud expéditeur.
Paquet TCP reçu	Données	Le nombre de paquets TCP reçus par le récepteur.
Le paquet TCP a chuté	Données	Le nombre de paquets TCP qui ont été supprimés.
Paquet TCP forward	Données	La somme du nombre de fois que chaque paquet TCP dans cette connexion a été transféré.
Énergie consommée		La quantité d'énergie consommée par tous les nœuds inhérents à la connexion.
Taux de livraison de paquets	Données	le rapport des paquets TCP reçus aux paquets TCP envoyés.
Délai moyen	Temps	Le délai moyen des paquets TCP reçus pendant la connexion.
Max hop	Routage	Nombre maximal de sauts pendant la connexion.
Houblon moyen	Routage	Nombre moyen de sauts pendant la connexion.
Débit	Données	la quantité de données reçues dans la connexion durée (kbps).

Tableau III. 2 : Descriptions des fonctionnalités ciblées.

Après l'application de l'anova on a découvrir que certains attributs (le débit et le délai moyen) parmi les 14 collectés dans le tableau 3.1 sont plus persistant que les autres et ces derniers sont nommés les attributs ciblés.

4. Script R détection :

Dans cette partie nous avons présenté notre script de détection programmé avec le langage R, ce script est illustré dans la figure III.3.

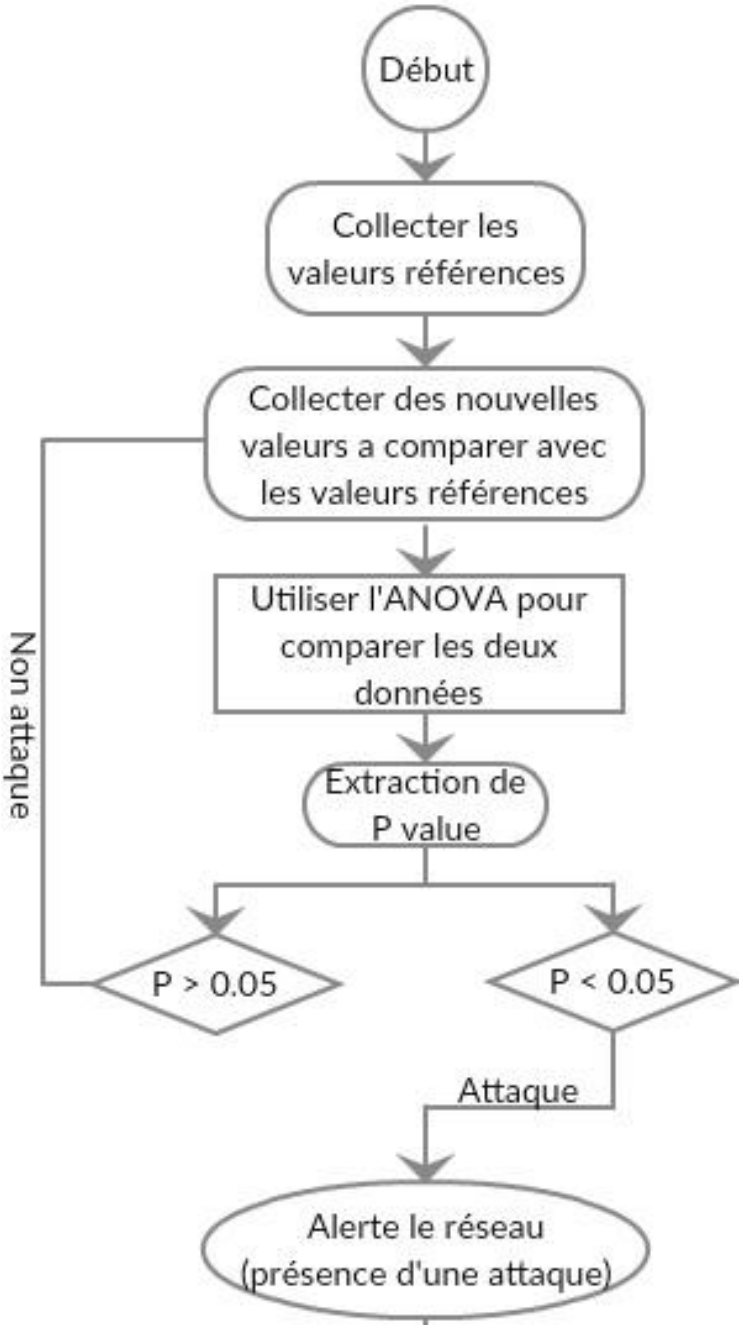

Figure III. 3 : Script R de détection.

5. Résultats et interprétations :

5.1 Métriques :

L'évaluation de la qualité de notre système IDS est faite avec les quatre éléments de base qui sont : les vrais positifs (VP), i.e. les anomalies effectivement détectées, les faux négatifs (FN), i.e. les anomalies non-détectées, les faux positifs (FP), i.e. les fausses alarmes et les vrais négatifs (VN). On peut les résumer sous la forme d'une matrice de confusion (Table 3.3).

	Prédit positif	Prédit négatif
Vrais positifs	VP	FN
Vrais négatifs	FP	VN

Tableau III. 3 : Matrice de confusion pour un problème de classification à 2 classes.

Pour mesurer le pourcentage d'échantillons correctement classés, on utilise l'incertitude (1) (i.e. le ratio de vrais positifs et vrais négatifs sur le nombre total d'échantillons). Cependant, il peut fausser l'évaluation lorsque le nombre de vrais négatif est considérablement supérieur au nombre de vrais positifs. C'est pourquoi l'on utilise d'autres métriques en complément.

$$\textbf{Incertitude : I} = \frac{\textbf{VP+FN}}{\textbf{VP+VN+FP+FN}} \quad \textbf{(1)}$$

Le premier critère est la précision (2), i.e. le ratio entre le nombre de varies anomalies et le nombre d'échantillons classés par l'algorithme comme des anomalies. Plus la valeur est petite et plus il y a de faux positifs. Le second critère est le rappel (3), qui est le pourcentage d'anomalies correctement classées en tant que telles parmi toutes les vraies anomalies. Ainsi, plus le rappel est petit et plus il y a d'anomalies non détectées.

$$\textbf{Précision : P} = \frac{\textbf{VP}}{\textbf{VP + FP}} (2) \qquad \textbf{Rappel : R} = \frac{\textbf{VP}}{\textbf{VP + FN}} \quad (3)$$

5.2 Résultats et analyses :

L'objectif principal de cette implémentation est de détecter les attaques DoS afin de déterminer l'efficacité, les points faibles ainsi que les perspectives envisagées de cette méthode proposée.

Sur la base de 100 simulations déployées, nous avons obtenu des résultats qu'ils sont présentés dans les tables suivantes :

Blackhole :

	Prédit positif	Prédit négatif
Vrais	6%	14%
Faux	16%	64%

Tableau III. 4 : Taux de détection de l'attaque Blackhole.

Flood :

	Prédit positif	Prédit négatif
Vrais	9%	26%
Faux	13%	52%

Tableau III. 5 : Taux de détection de l'attaque Flood.

DoS :

	Prédit positif	Prédit négatif
Vrais	28%	23%
Faux	12%	37%

Tableau III. 6 : Taux de détection de l'attaque DoS.

Nous avons atteint 70% de précision dans le scénario de Blackhole, 61% de précision dans le scénario Hello-Flood, et 65% de précision dans le scénario DoS. Les performances de notre système sont plutôt acceptables, mais pas assez fiables parce que nous pensons qu'il existe des facteurs que nous devons prendre en considération afin d'avoir une précision suffisante. Nous pensons aussi que notre méthode doit être réactive au changement de la variance qui est dû aux activités normales du réseau, l'exemple de la congestion, le changement de route dans le protocole AODV, ainsi que la densité du réseau.

Ces problèmes peuvent êtres résolues si on augmente le nombre d'attributs ainsi que le nombre des groupes a comparé parce que, L'ANOVA peut être appliquée à plusieurs groupes.

Métriques / Attaques	VP	VN	FP	FN	Incertitude	Précision	Rappel
Blackhole	6%	14%	16%	64%	0.7	0.27	0.09
Flood	9%	26%	13%	52%	0.61	0.41	0.15
DoS	28%	23%	12%	37%	0.65	0.7	0.43

Tableau III. 7 : Performance et précision.

On remarque que le niveau de précision dans la détection de l'attaque simulée DoS est plus grand que les autres, C'est parce que les données utilisées nous aident à identifier plus ce type d'attaque 'DoS'.

6. Conclusion :

Les attaques par déni de services existent depuis de nombreuses années et sont parfois médiatisées. Elles ont su se développer au fur et à mesure du développement des RCSF, tout en étant simples à mettre en œuvre. Bien qu'aujourd'hui, il existe des outils de détections et de protection contre le déni de service, ils sont toujours complexes à mettre en place avec une efficacité pas toujours optimale. Avoir une multitude d'outils ne suffit pas, il faut surtout être réactif lorsque l'on subit une attaque, mais aussi avoir une bonne politique de sécurité, non pas pour supprimer totalement les risques, car le risque zéro n'existe pas, mais au moins limiter l'impact d'une attaque de type déni de service face au service que l'on protège.

Dans ce chapitre, nous avons présenté notre système IDS basé sur un modèle statistique (ANOVA) utilisé pour comparer les moyennes d'échantillons pour RCSF, et nous avons présenté aussi l'ensemble des données est le résultat des données de simulation de prétraitement. Nous avons justifié tous les choix effectués sur le niveau de caractérisation des attaques, les attributs ciblés et l'algorithme à utiliser. Nous avons décrit notre étude expérimentale et commenté les résultats donnés.

Conclusion générale

Les attaques par déni de services existent depuis de nombreuses années et sont parfois médiatisées. Elles ont su se développer au fur et à mesure du développement des réseaux, tout en étant simples à mettre en œuvre. Les RCSF souffrent aussi de ce type d'attaque, et bien qu'aujourd'hui, il existe des méthodes de détections et de protection contre le déni de service, ils sont toujours complexes à mettre en place avec une efficacité pas toujours optimale et ceci à cause des spécificités des RCSF.

Dans ce travail, nous avons traité la problématique de la détection d'intrusion dans les (RCSF), nous avons également mentionné comment il est important de sécuriser ces réseaux face aux attaques DoS. Notre objectif est de déployer une solution allégée qui offre de hautes performances tout en respectant les spécificités et les limites des RCSF en termes d'énergie, de mémoire et de puissance de calcul.

La solution consistée à détecter les attaques par l'étude de l'impact des attaques DoS sur la variance d'un ou plusieurs attributs d'un RCSF. Pour ce faire, nous avons expérimenté de nombreux scénarios de simulation contenant plusieurs comportements ciblés. Ces comportements sont Normal, Blackhole, Hello-Flood et DoS. Ensuite on a déterminé quels sont les attributs les plus sensibles aux attaques DoS, pour les utiliser comme des indices révélateurs d'attaques DoS. Plus tard nous avons appliqué notre méthode de détection d'intrusion (IDS) qui est basée sur l'analyse de la variance ANOVA. Les performances obtenues sont plutôt acceptables, mais nécessitent des améliorations et des ajustements que nous recommandons pour des travaux ultérieurs.

Finalement, ce travail peut être enrichi en essayant d'autres configurations et d'autres protocoles de simulation.

Annexes

Annexe A

Comportements simulés NS2

1. Configuration et paramètres des nœuds

Set val (chan)	channel/wireless channel	; # type de canal
Set val (prop)	Propagation /TwoRayGround	; # modèle de propagation radio
Set val (netif)	Phy/ Wireless Phy	; #type d'interface réseau
Set val (mac)	Mac/802_1 1	; # Type MAC
Set val (ifq)	Queue/ DropTail / PriQueue	;# type de file d'attente
Set val (ll)	LL	;# type de couche de liaison
Set val (ant)	Antenna /OmniAntenna	;# modèle d'antenne
Set val (ifqlen)	200	;# paquet maximum en ifq
Set val (nn)	70	;# nombre de nœuds mobiles
Set val (rp)	AODV	;# type de protocole
Set val (x)	200	;# X dimension de la topographie
Set val (y)	200	;# Y dimension de la topographie
Set val (stop)	80	;# période de simulation

2. Initialisation d'objets et de fichiers de trace

Set ns [new Simulator]
Set tracefd [open Normal.tr w]
Set namtrace [open normal.nam w]
 # configurer un nouveau format de fichier de trace
$ns use-newtrace
$ns trace-all $tracefd
$ns namtrace-all-wireless $namtrace $val (x) $val(y)
Set dist (25m) 3.07645 e-07
Phy/WirelessPhy set CSThresh_ $ dist (25m)
Phy/WirelessPhy set RXThresh_ $ dist (25m)

3. Définition de la topographie et des valeurs aux paramètres configurés

```
# configurer un objet de topographie
Set topo                    [new Topography]
$topo load_flatgrid  $val (x) $val (y)
#Création du directeur général des opérations
create-god $val (nn)
# configurer les noeuds
$ns node-config-adhocRouting  $val (rp) \
-llType $val (ll) \
-macType $val (mac) \
-ifqType $val (ifq) \
-ifqLen $val (ifqlen) \
-antType $val (ant) \
-propType $val (prop) \
-phyType $val (netif) \
-channel [new $val (chan)] \
-topoInstanc e $topo \
-energyModel "EnergyModel" \
-initialEnergy 10 \
-txPower 0.33 \
-rxPower 0.1 \
-idlePower 0.05 \
-sleepPower 0.03 \
-agentTrace ON \
-routerTrace ON \
-macTrace OFF \
-movementTrace OFF \
```

4. Création et positionnement de nœuds

```
for  {set i 0} {$i< $val (nn)} {incri} {
Set mnode_ ($i) [$ns node]
$mnode_ ($i) set X_ [expr [expr [expr $i * 20] % 180] + 10]
$mnode_ ($i) set Y_ [expr [expr [expr $i * 70] / 280] * 10]
$mnode_ ($i) set Z_ 0.0}
$mnode (0) label " Sink "
for {set i 0} {$i< $val (nn)} {incri} {
$ns initial_node_pos $mnode ($i) 10}
```

5. Enregistrement du scénario de comportement normal

```
for {set i 0} {$i< $wn} {incri} {
Set random_node [expr int (rand ( ) _ 69) + 1]
#ajouter les nœuds dans les fichiers
puts $ nodes file "$random_node"
```

#Configurer une connexion TCP

```
Set tcp ($i) [new Agent/TCP]
$tcp ($i) set class 2
$tcp ($i) set {id [expr $ i + 1]
Set sink ($i) [new Agent/TCPSink]
$ns attach-agent $mnode ($random_node) $tcp_ ($i)
$ns attach-agent $mnode (0) $sink ($i)
$ns connect $tcp ($i) $sink ($i)
Set ftp ($i) [new Application /FTP]
$ftp ($i) attach-agent $tcp ($i)}

for {set i 0} {$i< $wn} {incri} {
Setstart_time [expr (rand ( ) + 2)]
Set stop_time [expr $start_time + 77]
puts $timefile" $start_time "
$ns at $start_time " $ftp ($i) start"
$ns at $stop_time " $ftp ($i) stop "}
```

6. Chargement du scénario de comportement normal

```
foreach line $nf {
```

#Configureruneconnexion TCP

```
set tcp ($j) [new Agent/TCP]
$tcp ($j) set class 2
$tcp ($j) set fid [expr $ j + 1]
set sink ($j) [new Agent/TCPSink]
$ns attach-agent $mnode($line) $tcp ($j)
$ns attach-agent $mnode(0) $sink ($j)
$ns connect $tcp ($j) $sink ($j)
if {$random_node == $line} {
set random_node [expr int (rand ( ) _ 69 ) + 1]}
set ftp_($j) [new Application/FTP]
$ftp_($j) attach-agent $tcp_($j)
incrj}
set j 0
foreach line $tf {
set start_time $line
set stop_time [expr $start_time + 77]
$ns at $start_time " $ftp ($j) start "
$ns at $stop_time " $ftp ($j) stop "
incr j}
```

7. Terminer la simulation et fermer les fichiers utilisés

 # Avertir les noeuds lorsque la simulation est terminée

```
for {set i 0} {$i< $val (nn)} {incri} {
$ns at $val (stop) "$mnode_($i) reset ; "}
```

 # fin nam et la simulation

```
$ns at $val (stop) " $ns nam-end-wireless $val (stop)"
```

```
$ns at [expr $val (stop) + 0.01] " puts \"end simulationn\" ; $ns halt "
proc stop {} {
global ns tracefdnamtrace
$ns flush-trace
close $ timefile
close $ nodesfile
close $ tracefd
close $namtrace}
```

8. Comportement normal

```
//Configuration et paramètres des nœuds
#définir le nombre des nœuds qui enverraient des données
Set wn [lindex $argv 0]
# Ouverture de fichiers en mode écriture pour sauvegarder les temps de démarrage et l'envoi
Set time file [open timef.txt w]
Set nodes file [open nodes.txt w]
// Initialisation d'objets et les fichiers trace
// Définition de la topographie et des valeurs aux paramètres configurés
// Création et positionnement de nœuds
// Enregistrement du scénario de comportement normal
//Terminer la simulation et fermer les fichiers utilisés
```

9. Blackholeattack

a. Le code ajouté à AODV.h

```
// Dans AODV classe agent de routage
boolmalicious ;
// Reste du code est ajouté à AODV.cc
```

9.2 TCL script

```
// Configuration et paramètres des nœuds
# définir le nombre des nœuds qui enverraient des données
Set wn [lindex $argv 0]
# Ouverture de fichiers en mode écriture pour sauvegarder les temps de démarrage et l'envoi
set time file [open timef.txt r]
set nodes file [open nodes.txt r]
set tf [read $ time file]
set nf [read $ nodes file]
set j 0
// Initialisation d'objets et les fichiers trace
// Définition de la topographie et des valeurs aux paramètres configurés
// Création et positionnement de nœuds
// Chargement scénario de comportement normal
set random_node [expr int (rand () _69) + 1]
$ns at 40.0 " [$mnode ($random_node) set ragent_] blackhole"
puts "$random_node"
// Terminer la simulation et fermer les fichiers utilisés
```

10. Hello flood attack

10.1. Le code ajouté à AODV.h

#définir FLOOD_INTERVAL 0.09

// dans les minuteries

class FloodTimer : public Handler{

public :

FloodTimer (AODV* a) : agent (a) {}

void handle (Event*) ;

private :

AODV *agent ;

Event intr;} ;

// Dans AODV classe agent de routage

friend class FloodTimer ;

// dans les routines TX paquet

voidFloodRREQ(nsaddr_t dst) ;

// Dans AODV: Agent Timers

boolflooder ;

FloodTimerftimer ;

// Reste du code est ajouté à AODV.cc

10.2 TCL script

// Configuration et paramètres des nœuds

 # définir le nombre des nœuds qui enverraient des données

setwn [lindex $argv 0]

 # Ouverture de fichiers en mode écriture pour sauvegarder les temps de démarrage et l'envoi

set time file [open timef.txt r]

set nodes file [open nodes.txt r]

set tf [read $ time file]

set nf [read $ nodes file]

set j 0

// Initialisation d'objets et les fichiers trace

// Définition de la topographie et des valeurs aux paramètres configurés

// Création et positionnement de nœuds

// Chargement scénario de comportement normal

set random_node [expr int (rand () * 69) + 1]

$ns at 40.0" [$mnode ($random_node) set ragent] flooder"

puts "$random_node"

// Terminer la simulation et fermer les fichiers utilisés

11. DoSattack

// Configuration et paramètres des nœuds

 # définir le nombre des nœuds qui enverraient des données

setwn [lindex $argv 0]

 # Ouverture de fichiers en mode écriture pour sauvegarder les temps de démarrage et l'envoi

set time file [open timef.txt r]

set nodes file [open nodes.txt r]

```
set tf [read $ time file]
set nf [read $ nodes file]
set j 0
// Initialisation d'objets et les fichiers trace
// Définition de la topographie et des valeurs aux paramètres configurés
// Création et positionnement de nœuds
// Chargement scénario de comportement normal
# créer une attaque DoS
set udp [new Agent/UDP]
$udp set fid $j
set null [new Agent/Nul l]
$ns attach-agent $mnode($random_node) $udp
$ns attach-agent $mnode(0) $null
set cbr [ new Application / Traffic /CBR]
$cbr set inter val 0.001
$cbr set random_ 1
$cbr set packetSize 1000
$cbr set maxpkts 300000
$cbr attach-agent $udp
$ns connect $udp $null
$ns at 40.0 " $cbr start "
$ns at 80.0 " $cbr stop "
// Terminer la simulation et fermer les fichiers utilisés
```

Annexe B

Script AWK

```
BEGIN {
        i=0;j=0;n=0;inc=0;
        for(i=1;i< 70;i++){
        start_time[i] = 80.000000000;}}
{
if (FILENAME == "nodes.txt"){
        cnxs++;
        sender_node[cnxs] = $0;}
else{
        state           =       $1;
        time            =       $3;
        flow            =       $39;
        pkt_size  =     $37;
        node_id   =     $9;
        level           =       $19;
        pkt_type  =     $35;
        packet_id=      $41;
        if(state != "N") {
        if ((state == "r" )&& (node_id == 0) && (level=="AGT")) {
                pkts_0_rcvd_size[flow][inc] = pkt_size;
                pkts_0_rcvd_time[flow][inc] = time;
                packet_end_time[flow][packet_id] = time;
                inc++;     }
        if ((state == "s" )&& (pkt_type == "tcp") && (level=="RTR")) {
                packet_start_time[flow][packet_id] = time;
                sender_node[flow] = node_id;
        if(time <start_time[flow] ) start_time[flow] = time;  }}}}
END {
for(i=1;i<=cnxs;i++) {
        if      (FILENAME == "Blackhole.tr"){file_name = "blackhole_" i ".csv"}
        else if (FILENAME == "DoS.tr")      {file_name = "dos_" i ".csv"}
        else if (FILENAME == "Flood.tr")    {file_name = "flood_" i ".csv"}
sum=0.00;
k= 0;
recvnum=0;
packet_duration=0.00;
delay[i]=0;
PROCINFO["sorted_in"] = "@ind_num_asc";
```

```
for (j in pkts_0_rcvd_size[i]){
        packet_0_end_time[i][k] = pkts_0_rcvd_time[i][j];
        rcvd_pkts_size[i][k]    = pkts_0_rcvd_size[i][j];
        k++;}
for ( j in packet_0_end_time[i]) {
        tot_rcvd_pkts_size[i][int(j/5)] += rcvd_pkts_size[i][j];
        tot_duration[i][int(j/5)] = packet_0_end_time[i][j];}
        Start_t = start_time[i];}
printf("et,")                   > (file_name);
printf("Average_delai,")        > (file_name);
printf("Throughput,\n")         > (file_name);
for (j in tot_duration[i]) {
for ( n in packet_end_time[i] ) {
        start = packet_start_time[i][n];
        end = packet_end_time[i][n];
if ((start >= Start_t) && (start <tot_duration[i][j])){
        packet_duration = end - start;
if ( packet_duration> 0 )
        {sum += packet_duration; recvnum++;}}}
if (recvnum != 0) delay[i]=sum/recvnum;
        Delai_time = tot_duration[i][j] - Start_t;
        printf("%.4f,", tot_duration[i][j])                                 >> (file_name);
        printf("%.9f,",delay[i])                                            >> (file_name);
        printf("%.4f\n", (tot_rcvd_pkts_size[i][j]*8)/Delai_time/1024)      >> (file_name);
        Start_t = tot_duration[i][j];
        sum = 0;
        recvnum=0;
        delay[i]=0; }}
}
```

Annexe C

Script R

// 1) Initialisation des variable TP, TN, FP, FN

```
B_tp<- 0 ;B_tn<- 0 ; B_fp<- 0 ; B_fn<- 0 ; F_tp<- 0 ;F_tn<- 0 ; F_fp<- 0 ; F_fn<- 0  ; D_tp<- 0 ;D_tn<- 0 ; D_fp<- 0 ;
D_fn<- 0

save (B_tp,B_tn,B_fp,B_fn,file="/home/abdi/Bureau/abdi/variable_blackhole.RData")

save (F_tp,F_tn,F_fp,F_fn,file="/home/abdi/Bureau/abdi/variable_flood.RData")

save (D_tp,D_tn,D_fp,D_fn,file="/home/abdi/Bureau/abdi/variable_dos.RData")
```

// 2) Script R pour calculer l'Anova

```
args<- commandArgs(TRUE)   // récupérer l'argument entrer

lenght_args<- nchar(args)              // calculer longueur de l'argument

chemin1     <- "/home/abdi/Bureau/abdi/"

chemin2     <- paste(chemin1, args, sep="") // le chemin de fichier entrer

Data        <- read.csv(chemin2) // ouvrir le fichier csv

library(MASS)

aed<- nrow(Data)

if (aed> 15) {
```

 # récupérer les variables qu'on a initialisé dans le 1 er script

```
if(lenght_args> 14){

   load("/home/abdi/Bureau/abdi/variable_blackhole.RData")

}else if (lenght_args> 10){

   load("/home/abdi/Bureau/abdi/variable_flood.RData")

}else {

   load("/home/abdi/Bureau/abdi/variable_dos.RData")}

j<- 10 ; p <- 1
```

```
Nbr_Boucle<- nrow(Data)/10 // calculer le Nombre des lignes de fichier principal

qq<- as.integer(Nbr_Boucle)

if (Nbr_Boucle<= qq){qq<- qq -1}

for (t in 1:qq) {

d <- j+1;e <- j+10

mm<- 1 ; Data1 <- Data[(1:10),] ; Data1 <- cbind(Data1,mm) // ajouter une valeur ‘1’ a les données

mm<- 2 ; Data2 <- Data[(d:e),]  ; Data2 <- cbind(Data2,mm) // ajouter une valeur ‘2’ a les données

Data3 <- rbind(Data1, Data2)   // collecter les deux données (Data1 et Data2)

Finish_Time<- Data3[20,3]

lenght_ET<- nchar(Finish_Time)
```

 # Récupéré la dernier valeur du END TIME

```
while(lenght_ET< 3){

ppp<- 20 –p ;Finish_Time<- Data3[ppp,3] ; p <- p+1

lenght_ET<- nchar(Finish_Time) }
```

 # Appliquer le test Anova

```
Data4 <- lm(cbind(Average_delai,Throughput) ~ mm, data=Data3)

#Data4 <- lm(Average_delai ~ mm, data=Data3)

#Data4 <- lm(Throughput ~ mm, data=Data3)

Annova<- anova(Data4)

#Pr <- Annova[1,5]

Pr <- Annova[2,6]// Récupéré la valeur P value du table anova

Data1 <- Data1[,-6]

Data2 <- Data2[,-6]

Data3 <- Data3[,-6]
```

 # Tester est ce que ilya une attaque ou pas

```
if (Pr<= 0.05) {

    if (Finish_Time<45){
```

```
if (lenght_args>14){
B_fp<- B_fp +1 // Incrimenter la valeur de B_fp (faux positif du Blackhole)
}else if (lenght_args> 10){
F_fp<- F_fp +1 // Incrimenter la valeur de F_fp (faux positif du Flood)
}else {
D_fp<- D_fp +1 // Incrimenter la valeur de D_fp (faux positif du DoS)}}
else {
if (lenght_args> 14){
B_tp<- B_tp +1 // Incrimenter la valeur de B_tp (true positif du Blackhole)
}else if (lenght_args> 10){
F_tp<- F_tp +1 // Incrimenter la valeur de F_tp (true positif du Flood)
}else {
D_tp<- D_tp +1 // Incrimenter la valeur de D_tp (true positif du DoS)}}
}else {
    if (Finish_Time<45){
        if (lenght_args>14){
B_fn<- B_fn +1
}else if (lenght_args> 10){
F_fn<- F_fn +1
}else {
D_fn<- D_fn +1 }}
        else {
         if (lenght_args>14){
B_tn<- B_tn +1
}else if (lenght_args> 10){
F_tn<- F_tn +1
}else {
D_tn<- D_tn +1}}}
```

```
j<- j+10}
```

 # Calculer les pourcentages de TP,TN,FP,FN

```
if (lenght_args>14){

  total <- B_tp+B_tn+B_fp+B_fn

BL_tp<- (B_tp*100)/total

BL_tn<- (B_tn*100)/total

BL_fp<- (B_fp*100)/total

BL_fn<- (B_fn*100)/total

Vrai <- BL_tp + BL_fn

  Faux <- BL_tn + BL_fp
```

 # exporter les résultats dans un fichier

```
externecapture.output(print("tp"),B_tp,print("tn"),B_tn,print("fp"),B_fp,print("fn"),B_fn,print("p_tp"),BL_tp,print("p_tn"),
BL_tn,print("p_fp"),BL_fp,print("p_fn"),BL_fn,print("Vrai"),Vrai,print("Faux"),Faux,
file="/home/abdi/Bureau/abdi/Resultat_blackhole.txt", append=FALSE)

save(B_tp,B_tn,B_fp,B_fn,file="/home/abdi/Bureau/abdi/variable_blackhole.RData")

}else if (lenght_args> 10){

  total <- F_tp+F_tn+F_fp+F_fn

FL_tp<- (F_tp*100)/total ;FL_tn<- (F_tn*100)/total ;FL_fp<- (F_fp*100)/total ;FL_fn<- (F_fn*100)/total

Vrai<- FL_tp + FL_fn; Faux <- FL_tn + FL_fp
capture.output(print("tp"),F_tp,print("tn"),F_tn,print("fp"),F_fp,print("fn"),F_fn,print("p_tp"),FL_tp,print("p_tn"),FL_tn,print
("p_fp"),FL_fp,print("p_fn"),FL_fn,print("Vrai"),Vrai,print("Faux"),Faux, file="/home/abdi/Bureau/abdi/Resultat_flood.txt",
append=FALSE) save(F_tp,F_tn,F_fp,F_fn,file="/home/abdi/Bureau/abdi/variable_flood.RData")

}else {

  total <- D_tp+D_tn+D_fp+D_fn

DL_tp<- (D_tp*100)/total  ;DL_tn<- (D_tn*100)/total  ;DL_fp<- (D_fp*100)/total  ;DL_fn<- (D_fn*100)/total

Vrai <- DL_tp + DL_fn;Faux <- DL_tn + DL_fp

capture.output(print("tp"),D_tp,print("tn"),D_tn,print("fp"),D_fp,print("fn"),D_fn,print("p_tp"),DL_tp,print("p_tn"),DL_tn,pr
int("p_fp"),DL_fp,print("p_fn"),DL_fn,print("Vrai"),Vrai,print("Faux"),Faux,
file="/home/abdi/Bureau/abdi/Resultat_dos.txt", append=FALSE)

save(D_tp,D_tn,D_fp,D_fn,file="/home/abdi/Bureau/abdi/variable_dos.RData")}}
```

Références

[1] Sukun Kim, Shamim Pakzad, David E. Culler, James Demmel, Gregory Fenves, Steve Glaser, and Martin Turon. Wireless sensor networks for structural health monitoring. In Andrew T. Campbell, Philippe Bonnet, andJohn S. Heidemann, editors, *SenSys*, pages 427–428. ACM, 2006.

[2] Matt Welsh. Deploying a sensor network on an active volcano. In *USENIX Annual Technical Conference, General Track*. USENIX, 2006.

[3] David Malan, Thaddeus Fulford-Jones, Matt Welsh, and Steve Moulton. Codeblue: An ad hoc sensor network infrastructure for emergency medical care. In *International Workshop on Wearable and Implantable Body SensorNetworks*, april 2004.

[4] D. W. Carman, P. S. Krus, and B. J. Matt. Constraints and approaches for distributed sensor network security. In *Technical Report 00-010, NAI Labs,Network Associates, Inc., Glenwood, MD*, 2000.

[5] Sun, B., Osborne, L., Xiao, Y., and Guizani, S. Intrusion detection techniques in mobile ad hoc and wireless sensor networks. IEEE Wireless Communications I.6, 5
(2007). 2, 24

[6] Butun, I., Morgera, S. D., and Sankar, R. A surveyof intrusion detection systems in wireless sensor networks. IEEE Communications Surveys&Tutorials 16, 1 (20I.6), 266{282. 2, 16, 22

[7] Antoine-Santoni Thierry, Santucci Jean Francois, de Gentili Emmanuelle,and Costa Bernadette. Using wireless sensor network for wildfire detection.a discrete event approach of environmental monitoring tool. In *Environment Identities and Mediterranean Area, 2006. ISEIMA '06. First international Symposium on*, 2006.

[8] Yacine Younes, "Minimisation d'énergie dans un réseau de capteurs", Mémoire de Master, Département d'Informatique, Université Mouloud Mammeri de Tizi-Ouzou, Septembre 2012.

[9] Tinyos. http:/www.tinyos.net/, 2010

[10] Dunkels, A., B. Grönvall, et T. Voigt. Contiki: a Lightweight and Flexible Operating System for Tiny Networked Sensors. In Proceedings of the First IEEE Workshop on Embedded Networked Sensors, pages 455-462, Tampa, Florida, USA, 2004.

[11] B. Jung and G. Sukhatme, “Multi-target tracking using a mobile sensor network”. In Proceedings of the IEEE International Conference on Robotics and Automation, May 2002.

[12] C. Chong and S. Kumar, “Sensor networks: Evolution, opportunities, and challenges” Proceedings of IEEE, August 2003.

[13] E. M. Petriu, N. D. Georganas, D. C. Petriu, D. Makrakis, and V. Z. Groza, “Sensor-based information appliances”. IEEE Information and Measurement Magazine, December 2000.

[14] B. Sibbald, “Use computerized systems to cut adverse drug events”. Canadian Medical Association Journal, June 2001.

[15] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, “Impact of Radio Irregularity on Wireless Sensor Networks”. MobiSys 2004.

[16] ALERT Systems. http://www.alertsystems.org/.

[17] N. Noury, T. Herve, V. Rialle, G. Virone, E. Mercier, G. Morey, A. Moro, and T. Porcheron, “Monitoring behavior in home using a smart fall sensor”. IEEE-EMBS SpecialTopic Conference on Microtechnologies in Medicine and Biology, October 2000.

[18] BouabdellahKechar, « Problématique de la consommation de l’énergie dans les réseaux de capteurs sans fil », Séminaire LIUPPA, Université de Pau et des Pays de l’Adour, 14 Octobre 2007.

[19] Lyes Khelladi, NadjibBadache « Les réseaux de capteurs: état de l’art », Rapport de recherche, Algérie, Février 2004.

[20] Wikipedia. \tinyos". site web, 2008. http://en.wikipedia.org/wiki/TinyOS.

[21] Contiki. web site. http ://www.sics.se/contiki.

[22] Adam Dunkels, Bj□ornGr□onvall, and Thiemo Voigt. Contiki { a lightweight and exible operating system for tiny networked sensors. IEEE EmNetS-I, November 2004. Swedish Institute of Computer Science.

[23] Bryan Parno, Adrian Perrig, and Virgil D. Gligor. Distributed detection ofnode replication attacks in sensor networks. In IEEE Symposium on Securityand Privacy, pages 49–63. IEEE Computer Society, 2005.

[24] Xun Wang, Wenjun Gu, Kurt Schosek, Sriram Chellappan, and Dong Xuan.Sensor network configuration under physical attacks. In Xicheng Lu andWeiZhao, editors, ICCNMC, volume 3619 of Lecture Notes in Computer Science,pages 23–32. Springer, 2005.

[25] C. Hartung, J. Balasalle, and R. Han. Node compromise in sensor networks: The need for secure systems. Technical Report CU-CS-988-04, Departmentof Computer Science, University of Colorado at Boulder, 2004.A.D. Wood and J.A. Stankovic. Denial of services in sensor networks. IEEEComputer, October 2002.

[26] Chris Karlof and David Wagner. Secure routing in wireless sensor networks:attacks and countermeasures. Ad Hoc Networks, 1(2-3):293–315, 2003.

[27] Frank Stajano and Ross J. Anderson. The resurrecting duckling: Security issues for ad hoc wireless networks. In Bruce Christianson, Bruno Crispo, James A. Malcolm, and Michael Roe, editors, Security Protocols Workshop,volume 1796 of Lecture Notes in Computer Science, pages 172–194. Springer,1999.

[28] J. Newsome, E. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks:analysis& defenses. In Information Processing in Sensor Networks, 2004. IPSN 2004. Third International Symposium on, pages 259–268, 2004.

[29] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Wormhole attacks in wireless networks. IEEE Journal on Selected Areas in Communications,24(2):370–380, 2006. [30] A.D. Wood and J.A. Stankovic. Denial of services in sensor networks. IEEE Computer, October 2002.

[30] J. P. Walters, Z. Liang, W. Shi, and V. Chaudhary, “Wireless Sensor Network Security: A Survey ”, Department of Computer Science Wayne State University.

[31] H. Krawczyk and M. Bellare and R. Canetti, "HMAC : Keyed-Hashing for Message Authentication", RFC 2104, 1997, February.