République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université A. Mira de Béjaia Faculté des Sciences Exactes
Département d'Informatique

Mémoire de fin de cycle

En vue de l'obtention du diplome Master recherche en Informatique

Option : Réseaux et Sécurité

Analyse des graphes d'attaques par la théorie des jeux dans les VANETs

Réalisé par :
MEZIANE Taous

Encadré par :
Mme. HAMZA Lamia

Soutenu le 02/07/2024, Devant le jury composé de :

Mme. Cherifi Feriel : M.C.B U. A/Mira Bejaia -Présidente
Mme. Bachiri Lina : M.C.B U. A/Mira Bejaia - Examinatrice

Année Universitaire : 2023/2024

⋆⋆ *Remerciements* ⋆⋆

Je remercie Allah le tout Puissant qui m'a donné la force et la volonté d'accomplir ce travail.

Je tiens à remercier docteur ***Lamia HAMZA****, pour ses qualités humaines et professionnelles, de m'avoir fourni aide, assistance et disponibilité, judicieux conseils et ses remarques pertinentes jusqu'à l'aboutissement de ce modeste travail.*

Je remercie les membres de jury, Mme. ***CHERIFI*** *et Mme.* ***BACHIRI*** *d'avoir accepté de corriger mon mémoire de fin de cycle et apporté les corrections necessaires afin d'améliorer la qualité du manuscrit.*

Je suis extrêmement reconnaissante envers ma mère et ma famille qui m'ont toujours soutenue tout au long de mon parcours.

Mes vifs remerciements et ma profonde gratitude sont adressés à tous mes amis(es), qui m'ont épaulé durant mon parcours, tant par leurs conseils que par leurs encouragements.

Mes très sincères remerciements vont aussi à tous les enseignants et enseignantes qui ont contribué à notre formation durant notre cursus.

⋆⋆ *Dédicace* ⋆⋆

Je dédie ce modeste travail :
À ma chère maman, ma source de courage
et le fil d'espoir qui illumine mon chemin. À toi, ma moitié.
À mon chère père.
À mes complices Amel et Wassim.
À toute ma famille.
À tous mes amis(es).
À Tout ceux qui m'aiment et que j'aime.

Taous MEZIANE

TABLE DES MATIÈRES

Table des matières **i**

Liste des tableaux **iv**

Liste des figures **v**

Liste des acronymes **vi**

Introduction générale **1**

1 Systemes de transport intelligents **2**
1.1 Introduction . . . 3
1.2 Definitions . . . 3
1.2.1 VANETs . . . 3
1.2.2 Internet des objets . . . 3
1.2.3 Système de transport intelligent . . . 3
1.3 Présentation des STI . . . 3
1.3.1 Historique et évolution des STI . . . 4
1.3.2 Objectifs des STI . . . 4
1.3.3 Composants clés des STI . . . 4
1.3.4 Technologies habilitantes . . . 6
1.3.5 Architecture des STI . . . 8
1.4 Présentation des VANETs . . . 9
1.4.1 Composants des VANETs . . . 9
1.4.2 Classification des Communications dans les VANETs . . . 9
1.5 Sécurité dans les VANETs . . . 10
1.6 Attaques dans les VANETs . . . 10
1.6.1 Attaque par déni de service . . . 10
1.6.2 Attaque par déni de service distribué . . . 11
1.6.3 Attaque black hole . . . 11
1.6.4 Attaque de type "wormhole" . . . 11
1.6.5 Attaque par illusion . . . 11
1.6.6 Le brouillage du système GPS . . . 11

1.7 Conclusion . . . 12

2 Etat de l'art **13**
2.1 Introduction . . . 14
2.2 Machine learning . . . 14
2.2.1 Apprentissage supervisé . . . 14
2.2.2 Apprentissage non-supervisé . . . 15
2.2.3 Apprentissage par renforcement . . . 15
2.3 Apprentissage profond . . . 15
2.4 ML et STI . . . 16
2.4.1 Intégration du ML à l'interieur des STI . . . 16
2.4.2 Éxploitation du ML par les tâches des STI . . . 16
2.5 ML dans la sécurisation des VANETs . . . 17
2.5.1 Critères d'évaluation . . . 18
2.5.2 Travaux antérieur . . . 18
2.6 Théorie des jeux dans les VANETs . . . 20
2.6.1 Définition d'un jeu . . . 20
2.6.2 Éléments constitutifs d'un jeu . . . 20
2.6.3 Types des jeux . . . 21
2.6.4 Jeux en forme normale . . . 21
2.6.5 Jeux en forme extensive . . . 22
2.6.6 Jeux stochastiques . . . 23
2.6.7 Méthodes de résolution . . . 24
2.6.8 Travaux antérieur . . . 25
2.7 ML et théorie des jeux dans les VANETs . . . 26
2.8 Classification . . . 26
2.9 Conclusion . . . 29

3 Propositions **30**
3.1 Introduction . . . 31
3.2 Proposition 1 : Théorie des jeux pour l'analyse des graphes d'attaques . . . 31
3.2.1 Cas d'étude . . . 31
3.2.2 Modélisation . . . 32
3.2.3 Forme normale du jeu . . . 37
3.2.4 Analyse du graphe d'attaques . . . 38
3.2.5 Élimination itérée des stratégies dominées . . . 38
3.2.6 Coût des chemins . . . 44
3.2.7 Evaluation . . . 47
3.3 Proposition 2 : Théorie des jeux et Machine learning pour l'analyse des graphes d'attaques . . . 47
3.3.1 Cas d'étude . . . 48
3.3.2 Modélisation . . . 48
3.3.3 Forme normale du jeu . . . 51
3.3.4 Analyse du graphe d'attaques . . . 52
3.3.5 Identification des clusters . . . 53
3.3.6 Analyse des clusters . . . 54
3.3.7 Identification des vulnérabilités prioritaires . . . 58
3.3.8 Evaluation . . . 59
3.4 Discussion . . . 59
3.5 Conclusion . . . 60

Conclusion générale et perspectives **61**

Bibliographie **62**

LISTE DES TABLEAUX

3.1 Exemples de vulnérabilités dans les VANETs 32
3.2 Tableau des coûts de stratégies d'attaques 35
3.3 Tableau des coûts de stratégies de défense. 35
3.4 Forme normale du jeu 38
3.5 Tableau des degrés de chaque vulnérabilité. 45
3.6 Exemples de vulnérabilités(Véhicule/RSU) 49
3.7 Tableau des coûts de stratégies d'attaques 50
3.8 Tableau des coûts de stratégies de défense. 50
3.9 Forme normale du jeu 52
3.10 Tableau des degrés de chaque vulnérabilité. 59

LISTE DES FIGURES

1.1 Historique des projets STI 5
1.2 Véhicule intelligent 5
1.3 Systemes de transport intelligents [6]. 6
1.4 Architecture de l'Internet des objets 8

2.1 Apprentissage par renforcement [12] 15
2.2 Interaction entre ML et STI 16
2.3 Forme extensive de l'exemple [36]. 23
2.4 Diagramme de classification des travaux étudiés 28

3.1 exemple d'attaque DDoS dans un reseau VANET. 31
3.2 Graphe d'attaque. 44
3.3 Graphe d'attaque aprés la suppression de V10. 46
3.4 Graphe d'attaque aprés la suppression de V9. 46
3.5 exemple d'attaque DDoS dans la communication véhicule RSU. 48
3.6 Graphe d'attaques correspondant à la topologie du réseau étudiée. 52
3.7 Données des coût de stratégies 53
3.8 Normalisation des données 53
3.9 Application Kmeans 53
3.10 Diagramme des clusters. 54
3.11 Graphe d'attaques aprés la suppression de V4 et V5. 58

LISTE DES ACRONYMES

AU Application Units

DDOS Distributed Denial Of Service

DOS Denial Of Service

DSRC Dedicated Short Range Communications

GPS Global Positioning System

IA Intelligence Artificielle

IoT Internet Of Things

MANET Mobile Ad Hoc Network

ML Machine Learning

PDA Personal Digital Assistance

RFID Radio Frequency IDentification

SL Supervised Learning

STI Systèmes de Transport Intelligents

UL Unsupervised Learning

V2I Vehicle-to-Infrastructure

V2V Vehicle-to-Vehicle

V2X Vehicle-to-Everything

VANETs Vehicular Ad Hoc Networks

INTRODUCTION GÉNÉRALE

Les Systèmes de Transport Intelligents (STI) répondent aux exigences croissantes du développement des transports. Ils intègrent les technologies de l'information, de la communication, de l'informatique, et autres, pour les appliquer dans le domaine des transports, créant ainsi un système intégré réunissant les personnes, les routes et les véhicules grâce à des technologies avancées de communication de données. Les STI permettent de mettre en place un système de gestion des transports étendu, multifonctionnel, en temps réel, précis et efficace. Ils recentrent l'attention des services, passant des gestionnaires de routes aux usagers de la route.

Comme tout système connecté, les STI, particulièrement les réseaux ad hoc véhiculaires, exposent les opérateurs de transport à des risques croissants en matière de cybersécurité. En effet, ces systèmes sont souvent interconnectés et échangent des données avec d'autres systèmes, des équipements variés et des réseaux hétérogènes, incluant notamment l'accès à Internet. Cette interconnexion accroît la vulnérabilité face aux attaques, augmentant ainsi le risque d'intrusions et de cyberattaques. Les conséquences de ces attaques peuvent être extrêmement graves. Il est donc primordial de mettre en place des mesures de sécurité solides pour protéger ces réseaux.

Dans notre projet de fin de cycle, nous nous penchons sur l'analyse des graphes d'attaques en vue de sécuriser les STI. En développant des approches basées sur la théorie des jeux et le machine learning, nous visons à simplifier la tâche des administrateurs dans la gestion de la sécurité de leur système.

Pour mener à bien notre projet de fin de cycle, nous avons structuré notre mémoire en trois chapitres :

Chapitre 1 : "Systèmes de transport intelligents" nous commençons ce mémoire par un premier chapitre portant sur les notions fondamentales des STI à savoir leur définition, leur architecture, etc.

Chapitre 2 : "Etat de l'art" comporte des notions de base sur le machine learning et la théorie des jeux que nous avons utilisé pour la réalisation de notre approche, et quelques travaux déjà réalisés sur la sécurisation des STI.

Chapitre 3 : "Proposition et implémentation" dans ce chapitre nous présentons nos contributions pour la sécurisation des STI. Nous abordrons la démarche suivi pour la réalisation de nos solutions, ensuite nous développerons les étapes de nos propositions et nous conclurons par des discussions.

À la fin, nous terminerons notre mémoire par une conclusion générale et perspectives.

CHAPITRE

1

SYSTEMES DE TRANSPORT INTELLIGENTS

1.1 Introduction

Les Systèmes de Transport Intelligents (STI) ont révolutionné la manière dont nous concevons, gérons et vivons la mobilité moderne. Ces systèmes, qui intègrent des technologies de l'information et des communications aux infrastructures de transport, offrent des avantages considérables en termes d'efficacité, de sécurité et de qualité de service. Cependant, tout progrès technologique s'accompagne de nouveaux défis et vulnérabilités, et les STI ne font pas exception.
Ce chapitre constitue le point de départ de notre exploration des enjeux liés à la sécurisation des STI. Il se penche sur un aspect critique de ce domaine, à savoir la sécurisation de ces systèmes. Alors que les STI offrent des avantages indéniables, ils sont exposés à diverses menaces, allant des attaques informatiques sophistiquées aux actes de vandalisme et de sabotage physiques. Remédier à ces menaces est essentiel pour préserver la sécurité, la fiabilité et la pérennité des systèmes de transport intelligents. Dans ce chapitre, nous allons exploré en details les STI et leur impact sur la mobilité urbaine.

1.2 Definitions

Les STI, les réseaux de véhicules ad hoc (VANET) et l'Internet des Objets(IoT) sont des piliers essentiels de l'infrastructure technologique contemporaine, travaillant de concert pour façonner l'avenir de la mobilité et de la connectivité. Les VANET, intégrés aux STI, facilitent la communication entre les véhicules et l'infrastructure routière, tandis que l'IoT fournit des capteurs et des dispositifs connectés, enrichissant ainsi les VANET de données cruciales pour une gestion efficace et une meilleure prise de décision dans les STI.

1.2.1 VANETs

Les VANETs (Vehicular Ad Hoc Networks) représentent une technologie innovante en matière de réseaux ad hoc mobiles (MANETs : Mobile Ad Hoc Network), dans lesquels les nœuds mobiles sont des véhicules intelligents munis d'équipements de pointe, tels que des ordinateurs, des radars, des systèmes de géolocalisation (GPS : Global Positioning System), divers types de capteurs et d'autres dispositifs réseau [1].

1.2.2 Internet des objets

L'Internet des objets (IoT : Internet Of Things) est définit comme étant un réseau d'objet physiques munis de capteurs, d'actionneurs et d'interfaces de communication, qui leur permettent de communiquer, de collecter et d'échanger des informations dans le but de réaliser des tâches spécifiques en vue d'améliorer la qualité de vie [2].

1.2.3 Système de transport intelligent

Un STI est l'ensemble de solutions basées sur la combinaison des technologies des télécommunications et de l'informatique, conçu et développé pour améliorer la gestion, la maintenance, la surveillance, le contrôle et la sécurité des transports [1].

1.3 Présentation des STI

Dans cette section dédiée aux STI, nous couvrirons brièvement leur histoire, leurs éléments constitutifs, leurs objectifs, les technologies sous-jacentes, ainsi que leur architecture fondamen-

tale. Cela permettra d'obtenir une vue d'ensemble complète de ces systèmes innovants.

1.3.1 Historique et évolution des STI

L'histoire des STI est marquée par une série de développements technologiques et d'initiatives visant à améliorer la gestion et la sécurité du transport [3].Voici un bref aperçu de l'historique des STI :

1. **Années 1960-1970 :**
   - Les débuts des systèmes de gestion de la circulation basés sur des capteurs et des signaux.
   - Développement des premiers systèmes de contrôle de feux de signalisation.
2. **Années 1980 :**
   - Lancement de projets pilotes pour les systèmes de transport intelligents.
   - Utilisation des premiers systèmes de gestion de la circulation assistés par ordinateur.
3. **Années 1990 :**
   - Introduction des systèmes de gestion du trafic intelligent pour améliorer la fluidité du trafic.
   - Développement de systèmes de gestion de flottes pour les transports en commun.
4. **Années 2000 :**
   - Expansion des systèmes de transport en temps réel pour les voyageurs, y compris l'information sur les horaires de transports publics.
   - Croissance de la connectivité véhicule-infrastructures.
5. **Années 2010 et au-delà :**
   - Adoption généralisée des technologies de véhicules autonomes et semi-autonomes.
   - Développement de systèmes de transport intelligents à l'échelle des villes et des régions.

La figure 1.1 résume l'histoire des STI jusqu'aux années 2000.

1.3.2 Objectifs des STI

Les STI sont configurés de manière à exploiter des unités en bordure de route et des équipements embarqués. Les fonctions de base de ce système sont les suivantes[4] :

- acheminement et échange d'informations contribuant à réduire le nombre d'accidents de la route.
- acheminement et échange d'informations contribuant à favoriser une conduite sûre.
- acheminement et échange d'informations contribuant à fluidifier les flux de circulation.
- améliorer la sécurité et le confort des passagers.

1.3.3 Composants clés des STI

Les principaux composants des STI sont :

1. **Les véhicules intelligents**
   Les STI sont le plus souvent associés aux véhicules intelligents, qu'ils disposent de technologies d'assistance au conducteur, qu'ils soient semi-autonomes ou même entièrement autonomes. Les véhicules intelligents sont un élément majeur des STI en raison du grand nombre de véhicules personnels sur la route. Alors que l'industrie des transports continue d'évoluer, les développements liés aux véhicules intelligents ont le plus grand impact sur

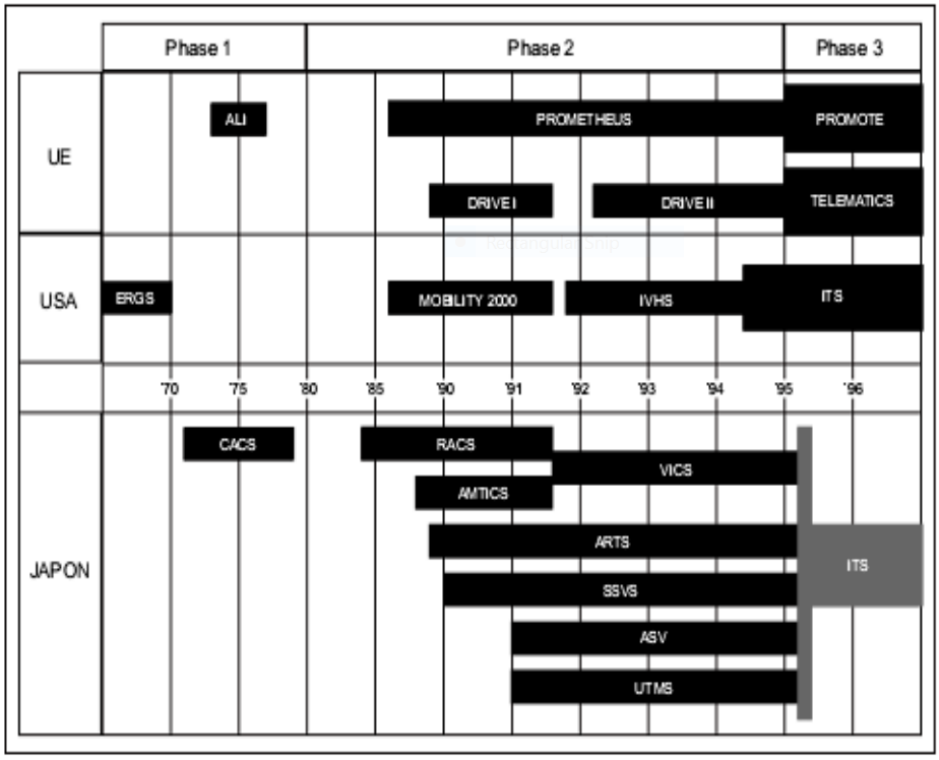

FIGURE 1.1 – Historique des projets STI

la manière dont nous voyageons et sur l'infrastructure sous-jacente des transports. Les véhicules intelligents peuvent créer des "convois" en utilisant la technologie de communication de véhicule à véhicule (V2V) pour améliorer l'efficacité des déplacements [5]. Cette

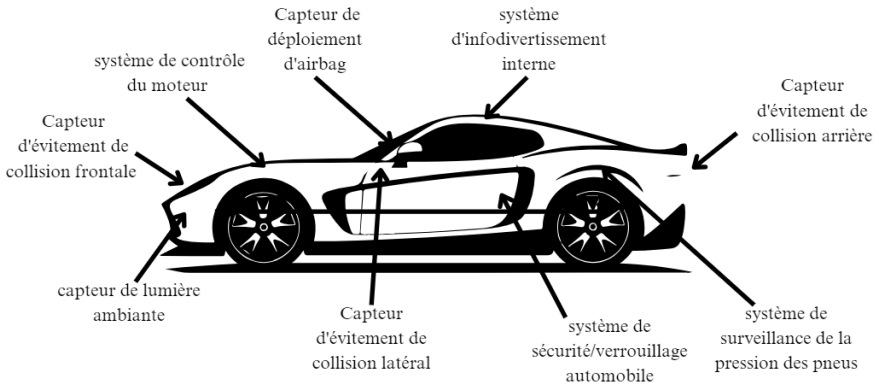

FIGURE 1.2 – Véhicule intelligent

communication et regroupement de véhicules a été rendue possible grâce à la recherche dans les réseaux ad hoc véhiculaires . La figure 1.2 illustre la structure d'un véhicule intelligent.

2. **Transports en commun**
   Les transports en commun urbains sont cruciaux et les STI promettent d'améliorer l'efficacité, notamment grâce à des arrêts de bus intelligents fournissant des horaires et des informations sur les retards, une optimisation des itinéraires en temps réel, et l'interaction avec l'IoT pour suivre les passagers.

3. **Dispositifs de l'IoT**
   Les smartphones permettent l'intégration avec les véhicules intelligents et facilitent la communication avec les piétons, une partie essentielle des STI. Les piétons utilisant les infrastructures de transport public peuvent recevoir des informations en temps réel sur la météo, la circulation, les incidents, etc., via leurs smartphones connectés au réseau de capteurs et de signaux des STI [5]. Les dispositifs IoT, tels que les capteurs et les micro-contrôleurs, sont de plus en plus utilisés pour collecter et traiter des données utiles pour les algorithmes de gestion de la circulation.

4. **Les contrôleurs**
   Les contrôleurs des STI sont les composants qui administrent, contrôlent ou modifient la dynamique du système de transport. Les contrôleurs au sein des STI traduisent la prise de conscience, la réponse et/ou l'action physique du système en fonction des données observées. Les exemples de contrôleurs comprennent les feux de circulation, les systèmes d'annonces de trafic, les panneaux de signalisation numériques et les aiguillages ferroviaires.
   La figure 1.3 illustre les composants des STI.

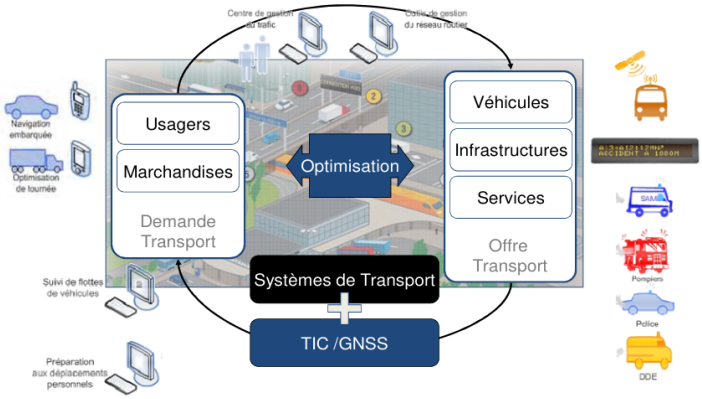

FIGURE 1.3 – Systemes de transport intelligents [6].

1.3.4 Technologies habilitantes

Dans cette section, nous présentant les technologies habilitantes des STI, qui sont essentielles pour recueillir, analyser et utiliser efficacement les données.

1. **La detection (Capteurs)**
   Les capteurs sont répartis dans l'ensemble des STI afin de recueillir des données pertinentes, allant du nombre de véhicules en attente à un feu de signalisation spécifique aux conditions de température et de précipitations dans une région donnée. Ces capacités de détection offrent au STI la possibilité de prendre des décisions éclairées et correctes pour améliorer l'efficacité et/ou la sécurité [7].

2. **Le Calcul**
   Le calcul joue un rôle essentiel dans les STI et est utilisé dans quasiment tous les éléments et processus au sein de ce système. Par exemple, les calculs cryptographiques nécessaires pour de nombreux protocoles de communication dans les STI exigent des microcontrôleurs ou des

ordinateurs embarqués pour effectuer ces opérations, garantissant ainsi la confidentialité des messages transmis sur le réseau. Le cloud computing a émergé comme une solution potentielle pour le traitement à grande échelle requis pour gérer les énormes quantités de données générées par les STI.

3. **Analyse de données**
   Dans les STI, la composante analytique est essentielle pour fournir l'intelligence au système. En utilisant diverses données, telles que les retards de circulation, le taux de congestion, le flux de piétons, etc., les STI sont capables d'extraire des informations sur l'état global du système, ce qui leur permet de prendre des décisions visant à réduire la congestion et à minimiser les retards [3]. En tirant parti des capacités offertes par le cloud computing, un réseau analytique hiérarchique peut être mis en place. Dans ce réseau, les décisions relatives à la circulation au niveau régional peuvent être prises par des serveurs d'analyse intermédiaires, tandis que des décisions à l'échelle du système continuent d'être prises par une autorité centrale de la circulation.

4. **La technologie V2X (Vehicle-to-Everything)**
   Représente une approche qui tire parti des réseaux de communication des STI pour diffuser et relayer des informations, telles que des alertes de sécurité, des données météorologiques, des informations sur la circulation et des instructions de navigation, entre les véhicules et divers composants des STI, et vice versa. V2X comprend à la fois la technologie V2V (Vehicle-to-Vehicle), où les véhicules communiquent entre eux, comme c'est le cas dans les réseaux VANETs, partageant des informations sur leur vitesse, leur position et les dangers potentiels sur la route, ainsi que la technologie V2I (Vehicle-to-Infrastructure), qui implique un échange sans fil bidirectionnel d'informations entre les véhicules et l'infrastructure routière, comprenant des lecteurs RFID(Radio Frequency IDentification) aériens, des caméras, des marquages de voie, des feux de signalisation, des lampadaires, des panneaux de signalisation et des horodateurs [5]. Lorsque des véhicules intelligents circulent sur les routes, les unités embarquées à l'intérieur de ces véhicules communiquent avec les unités en bord de route. Grâce à l'infrastructure V2X, une variété de signaux et de messages peuvent être diffusés ou reçus pour rendre les routes plus efficaces et plus sûres pour l'ensemble des usagers.

5. **Les réseaux de communication**
   Au sein des STI, les réseaux de communication jouent un rôle essentiel en reliant les composants STI entre eux et en facilitant la communication entre différentes applications. Ces communications permettent d'échanger des informations de sécurité entre les véhicules, améliorant ainsi la connaissance du trafic, des conditions routières et des dangers potentiels pour les véhicules intelligents. Le système repose sur le DSRC (Dedicated Short Range Communications) pour assurer la transmission fiable d'informations. Pour acheminer les données des unités en bord de route vers les serveurs de gestion du trafic, des réseaux de communication supplémentaires sont utilisés.

6. **Les contrôleurs de circulation intelligents**
   Les contrôleurs de circulation intelligents sont responsables de la mise en œuvre des décisions générées par les composants analytiques/intelligents des STI. Ils reçoivent des algorithmes de routage et des schémas de circulation adaptés en fonction des observations faites par divers éléments tels que les véhicules intelligents, les transports en commun et les dispositifs de l'IoT. Ils ne se limitent pas à réagir aux données observées, mais ils contribuent également à la collecte de données.

1.3.5 Architecture des STI

Les STI peuvent être considérés comme une sous-catégorie de l'IoT, et par conséquent, ils peuvent être développés en utilisant des approches et des architectures similaires. La Figure 1.4 illustre les contours architecturaux de la plupart des développements de l'IoT, et cette approche peut également être appliquée aux STI [8].

Donc les STI peuvent être envisagés comme un modèle stratifié. Dans ce modèle, quatre couches horizontales (perception, réseau, support, application) décrivent les diverses fonctionnalités propres à chaque niveau, tandis que deux couches transversales se concentrent sur la sécurité et la gestion des données.

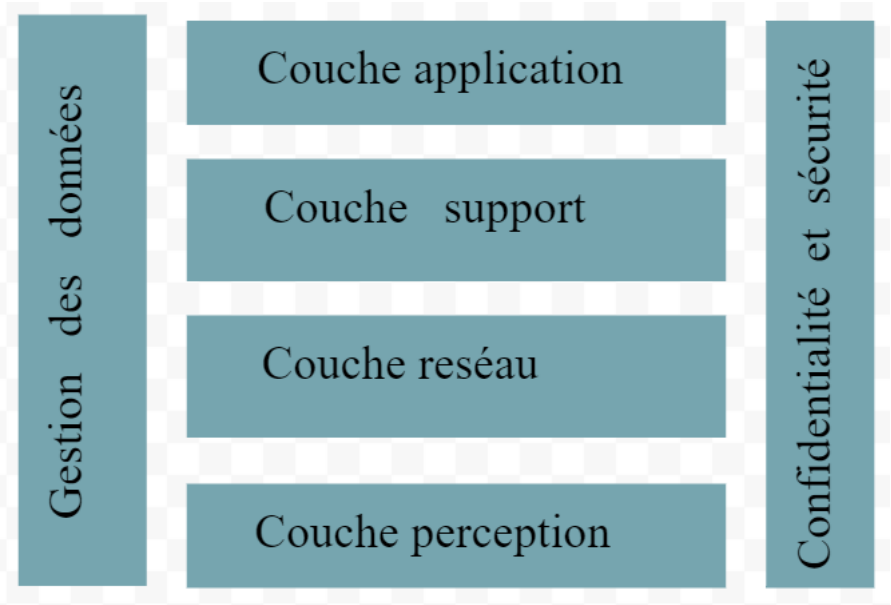

FIGURE 1.4 – Architecture de l'Internet des objets

1. **Couche perception :** intègre les smartphones des utilisateurs, les capteurs embarqués, et les équipements d'infrastructure.

2. **Couche réseau :** est en charge de la gestion du transfert des paquets depuis la source vers leur destination, y compris l'encapsulation qui forme ces paquets.

3. **Couche support :** inclut diverses technologies telles que le cloud computing, le calcul intelligent, le Fog Computing, et d'autres encore.

4. **Couche application :** englobe l'interaction avec l'utilisateur final, se manifestant par la présentation d'informations, la fourniture d'avertissements, voire l'activation de divers systèmes dans le véhicule.

5. **Couche gestion de données :** est une couche transversale qui interagit avec l'ensemble des couches précédentes. Elle a pour responsabilité le stockage et la gestion des données, ce qui inclut les opérations d'extraction, de transformation (ETL), la gestion des bases de données, et la gestion des entrepôts de données, entre autres.

6. **Couche Sécurité :** revêt une importance majeure dans cette architecture, car elle est chargée de garantir la sécurité des dispositifs de perception tels que les caméras et les capteurs. Son rôle consiste à assurer la sécurité de l'ensemble du processus, de l'acquisition initiale des données à leur traitement, leur transmission et leur diffusion ultérieure.

1.4 Présentation des VANETs

Dans cette section, nous explorerons en détail les principaux composants du réseau ad-hoc véhiculaire. Chaque élément essentiel joue un rôle spécifique et crucial dans la facilitation de la communication et de la sécurité au sein du réseau.

1.4.1 Composants des VANETs

1. **Unité embarquée (OBU : On Board Unit)** est un dispositif réseau, fixé sur les véhicules, et utilisé pour échanger des informations avec les OBUs des véhicules voisins ou avec les unités en bord de route. Les capteurs de l'unité de contrôle électronique (ECU : Electronic Control Unit) collectent des informations sur le véhicule et son environnement, puis l'unité d'application traite ces informations, génère des messages basés sur les informations recueillies et partage ces informations avec le véhicule voisin afin d'éliminer tout incident. L'OBU peut se connecter à Internet via les RSUs par radio DSRC ou par Hotspot et, en l'absence de ces deux options, les OBUs peuvent communiquer entre eux via un réseau cellulaire [9].
2. **Unité en bord de route (RSU : Road Side Unit)** agit comme une station de base ou un pont entre les véhicules et les services routiers fournis par le VANET. Les RSUs sont statiques par nature car elles ont une portée fixe, seuls les véhicules dans cette portée peuvent communiquer avec cette RSU. La fréquence et la distribution des RSUs dépendent principalement du protocole de communication utilisé. La RSU assiste l'autorité de confiance (TA) pour révoquer la communication d'un nœud autorisé vers des nœuds non autorisés ou malveillants [10].
3. **Autorité de confiance (TA : Trusted Authority)** est responsable de la délivrance des certificats numériques aux RSUs et aux véhicules, qui authentifient de manière unique le véhicule et la RSU dans le réseau ad-hoc véhiculaire. Pour isoler les véhicules non fiables, l'autorité de confiance maintient une liste des véhicules malveillants et diffuse continuellement cette liste dans le réseau, afin de prévenir la participation de ces véhicules au VANET en raison de leur comportement malveillant dans le passé [11].
4. **Unités d'application (AU : Application Units)** est un dispositif équipé dans le véhicule. La connexion entre l'AU et l'OBU peut être câblée ou sans fil, et dans certains cas, l'AU et l'OBU peuvent être combinés en une seule puce. L'AU gère les services fournis par le fournisseur, principalement dans les applications de sécurité et l'assistance numérique personnelle (PDA : Personal Digital Assistance).

1.4.2 Classification des Communications dans les VANETs

Les VANET sont très différents des MANETs, car les nœuds dans les MANETs sont libres de se déplacer de manière aléatoire alors que dans les VANETs, les véhicules ou nœuds suivent des chemins fixes comme les routes ou les autoroutes sur lesquelles ils sont libres de se déplacer. La communication dans les VANETs est catégorisée en quatre classes [10] :

— **Communication intra-véhicule**
Elle est très importante pour le conducteur car elle fournit des informations relatives

à la santé du véhicule, aux performances et à la fatigue et somnolence du conducteur, essentielles pour la sécurité du conducteur et du public.

— **Communication véhicule à véhicule (V2V)**
Elle permet la communication entre les véhicules, où les véhicules peuvent directement communiquer avec d'autres véhicules, ce qui implique l'envoi et la réception de messages.

— **Communication véhicule à infrastructure (V2I)**
Elle décrit la communication entre les véhicules et l'infrastructure, comme les feux de circulation ou les unités en bord de route (RSU). Les RSUs agissent comme des routeurs fixés sur le bord de la route et ayant une certaine hauteur, lorsqu'elles reçoivent des informations d'un véhicule, elles transmettent ces informations à la destination spécifiée. Les RSUs sont placées à une distance fixe, cette distance dépend de la limite de communication des dispositifs RSUs. Pour une communication efficace, le VANET préfère utiliser la norme IEEE 802.11.

— **Communication véhicule à large bande (V2B)**
Elle définit que les véhicules peuvent communiquer via certains canaux de communication à large bande sans fil comme HSPA (3G) ou Long Term Evolution (4G) car ces réseaux à large bande intègrent des informations supplémentaires sur le trafic et des données de surveillance utiles pour le suivi des véhicules et l'assistance active au conducteur [9].

1.5 Sécurité dans les VANETs

Dans cette section, nous présentons quelques attributs de sécurité pour les VANETs. En ce qui concerne la sécurité, ces trois attributs ne peuvent être ignorés.

— **Authenticité** est cruciale car elle authentifie que le message a été envoyé par un véhicule authentifié. Il est essentiel de garantir l'authenticité de l'expéditeur du message.

— **Disponibilité** garantit que le canal de communication et les ressources réseau doivent être disponibles même si le réseau est soumis à une attaque, sans affecter les performances du réseau.

— **Confidentialité** est obtenue en utilisant une technique cryptographique sur les messages. La confidentialité fait référence à une communication confidentielle. Lorsqu'un message est transmis, seul le destinataire authentifié peut décrypter ce message.

1.6 Attaques dans les VANETs

Dans cette section, nous allons explorer Certaines types d'attaques dans les VANETs et leurs effets.

1.6.1 Attaque par déni de service

Dans une attaque par déni de service (DOS : Denial Of Service), l'attaquant cible les services fournis par un fournisseur de services. Les utilisateurs légitimes ne peuvent pas accéder au service dans le réseau même si des ressources libres sont disponibles. L'attaquant perturbe le principal canal de communication. Ce type d'attaque est limité à la portée du fournisseur de services. Les attaquants peuvent atteindre cet objectif de deux manières différentes : au niveau de base, les attaquants submergent les ressources en envoyant de grands ensembles de requêtes. Ainsi, les ressources restent continuellement occupées à répondre à de telles requêtes frauduleuses et ne peuvent pas répondre aux demandes d'autres utilisateurs légitimes. Ces

attaques peuvent être intensifiées en envoyant un très grand nombre de requêtes pour saturer la communication. Ainsi, l'RSU ne peut pas gérer les demandes envoyées par l'OBU.

1.6.2 Attaque par déni de service distribué

L'attaque par déni de service distribuée (DDOS : Distributed Denial Of Service) est générée en orchestrant une attaque DOS de manière distribuée. Dans une attaque DDOS, plusieurs attaquants ciblent un ou plusieurs fournisseurs de services à partir de différentes localisations pour créer des perturbations dans l'utilisation des services fournis par le fournisseur de services. Dans cette attaque, un plus grand nombre de nœuds OBU malveillants sont impliqués, ce qui bloque les autres utilisateurs légitimes d'accéder aux services à partir d'un ou plusieurs RSU [11]. Les attaquants augmentent la latence de transmission inutile du réseau en envoyant des messages indésirables dans le réseau. La propagation de ce type d'attaque est très dangereuse pour les VANET.

1.6.3 Attaque black hole

Dans cette attaque, des nœuds malveillants dans un réseau créent une zone où le trafic réseau est redirigé ou les paquets de messages sont abandonnés. Les nœuds malveillants transmettent de fausses informations de routage et prétendent avoir une route optimale pour la destination afin d'attirer le nœud émetteur. Lorsque le nœud émetteur transmet ce paquet, les véhicules malveillants abandonnent ce paquet ou l'utilisent à leur avantage [9].

1.6.4 Attaque de type "wormhole"

Dans cette attaque, les voitures malveillantes reçoivent les paquets de données des voitures légitimes. Les voitures malveillantes créent un tunnel entre l'émetteur et le récepteur avec un nombre minimal de sauts, et enregistrent ce chemin dans la table de routage. Lorsque le nœud émetteur doit transmettre des données, il cherche le chemin de destination avec le nombre de sauts le plus faible et choisit celui créé par les nœuds malveillants. Il commence alors à transmettre les paquets de données par ce chemin. Comme la communication entre l'émetteur et le récepteur passe par ce tunnel malveillant, les nœuds malveillants peuvent soit abandonner les paquets, soit les écouter, les modifier ou les utiliser à leur avantage.

1.6.5 Attaque par illusion

L'attaquant dans cette attaque manipule volontairement les données de son véhicule ou les informations de trafic, puis transmet ces fausses informations aux véhicules voisins et aux RSUs. Dans un réseau VANET, le comportement des conducteurs dépend des messages d'alerte qu'ils reçoivent. Par conséquent, en recevant de faux messages d'alerte, les conducteurs peuvent modifier leur comportement, ce qui peut entraîner des accidents, des embouteillages ou réduire les performances du réseau en perturbant la topologie du réseau.

1.6.6 Le brouillage du système GPS

Cette attaque est également connue sous le nom d'attaque de falsification de position. Dans ce type d'attaque, l'attaquant tente de modifier l'identité de la position géographique actuelle et de produire de fausses informations à partir du système GPS en utilisant une telle technique. L'utilisateur cache ainsi sa position réelle au réseau et affiche une fausse position aux autres. Cette attaque peut être réalisée par un seul véhicule ou par un groupe de véhicules.

1.7 Conclusion

Dans ce premier chapitre, nous avons présenté une étude détaillée sur les systèmes de transport intelligent. notamment sa définition, son historique, ses objectifs, ses composantes, ses technologies et son architecture.
Nous avons exploré les composants principaux des VANET, tels que la communication V2V et V2I, les systèmes embarqués, et les technologies de communication sans fil, et expliqué leur rôle dans la facilitation de la communication et de la sécurité au sein du réseau.
Par ailleurs, nous avons examiné les aspects de la sécurité dans les VANET, en présentant les différentes menaces et attaques potentielles. Dans le chapitre suivant, nous examinerons quelques articles qui proposes un ensemble d'approches dédiées à la sécurisation des STI.

CHAPITRE

2

ETAT DE L'ART

2.1 Introduction

Les STI sont censés être un élément intégral de la planification urbaine et des futures villes intelligentes, contribuant à améliorer la sécurité routière et la circulation. Toutefois, cette intégration ne se fait pas sans poser divers défis, notamment en raison de l'ampleur de leur déploiement et de la diversité de leurs exigences en termes de qualité de service, ainsi que de la quantité massive de données qu'ils génèreront. Parallèlement à cette évolution, les techniques d'apprentissage automatique (ML : Machine learning) connaissent un essor significatif grâce à l'avancée de diverses technologies. Au cours des dix dernières années, on observe une tendance marquée à l'augmentation de l'utilisation du ML pour faciliter et optimiser les tâches des STI. Dans ce chapitre, nous allons aborder la manière dont les techniques du ML sont utilisées par les applications des STI et plus précisement les VANETs pour assurer une meilleure sécurité. Nous allons aussi présenter les travaux utilisant ML et théorie des jeux pour renforcer la sécurité des STI.

2.2 Machine learning

ML est une branche de l'intelligence artificielle (IA) qui se concentre sur le développement de techniques permettant aux ordinateurs d'apprendre à partir de données et d'améliorer leurs performances dans des tâches spécifiques sans être explicitement programmés. En utilisant des algorithmes et des modèles statistiques, le ML permet aux systèmes informatiques de reconnaître des patterns dans les données, de prendre des décisions autonomes et d'effectuer des prédictions. Dans cette section, nous nous concentrons sur les approches ML courantes, y compris l'apprentissage supervisé (SL : Supervised Learning), l'apprentissage non supervisé (UL : Unsupervised Learning), l'apprentissage par renforcement (RL : Reinforcement Learning) et l'apprentissage profond (DL : Deep Learning).

2.2.1 Apprentissage supervisé

Dans le cas de l'apprentissage supervisé, l'objectif est d'apprendre une estimation d'une fonction inconnue f, de correspondance entre les données d'entrées x et des données de sorties y telle que $y = f(x)$ [12].
Un jeu de données $D = (xi, yi)_{i=1}^{N}$ est un ensemble composé de N exemples de paires d'entrée et de sortie (xi, yi) [13]. On considère x comme un ensemble de caractéristiques d'entrée et y comme la valeur de sortie, qu'elle soit catégorielle ou continue.
Ce type d'apprentissage implique l'utilisation d'une partie des données D pour permettre au modèle d'apprendre à prédire la valeur de sortie avec une précision maximale en minimisant une fonction de coût via la fonction d'estimation $\hat{f}$. Une fois l'apprentissage terminé, le modèle peut prédire avec une erreur minimale la valeur estimée $\hat{y}$ pour toute nouvelle valeur de x.
Les algorithmes d'apprentissage supervisé se divisent en deux grandes catégories : la classification et la régression.

- **Les algorithmes de classification** apprennent à prédire une catégorie pour de nouvelles observations basées sur des données d'entraînement étiquetées, comme les machines à vecteurs de support (SVM :Support Vector Machine) et le Boosting Adaptatif (AdaBoost : Adaptive Boosting).
- **Les algorithmes de régression** travaillent sur des problèmes où la sortie est une valeur réelle ou continue, en tentant d'ajuster les données avec un meilleur hyperplan, tel que la régression linéaire (LR : Linear Regression) ou la régression par vecteurs de support (SVR : Support Vector Regression).

Certains algorithmes, comme les k plus proches voisins (k-NN : k-Nearest Neighbors), les forêts aléatoires (RF : Random Forest), et les arbres de régression boostés (BRT : Boosted Regression Trees), sont utilisés à la fois pour les problèmes de classification et de régression.

2.2.2 Apprentissage non-supervisé

En apprentissage non-supervisé, une différence majeure réside dans le jeu de données $D = (xi)_{i=1}^{N}$. Il n'est composé que de N exemples d'entrée x sans valeur de sortie yi [12]. C'est une méthode de découverte de connaissances basée sur les données, peut inférer une fonction décrivant la structure à partir de jeux de données contenant uniquement des données d'entrée non étiquetées. On distingue deux catégories d'algorithmes non supervisés :

— **Les algorithmes de regroupement** tels que le regroupement K-moyennes (K-Means Clustering), qui découvrent les regroupements inhérents dans les données.

— **Les algorithmes de réduction de dimension**, on trouve l'analyse en composantes principales (PCA : Principal Component Analysis) et l'analyse en composantes indépendantes (ICA : Independent Component Analysis), qui cherchent à trouver la meilleure représentation des données avec un nombre réduit de dimensions.

2.2.3 Apprentissage par renforcement

L'apprentissage par renforcement se distingue fondamentalement des autres formes d'apprentissage. Dans ce processus, un agent interagit avec son environnement, tel qu'illustré dans la Figure 2.1. En se basant sur ses observations, l'agent prend des décisions quant à ses actions.

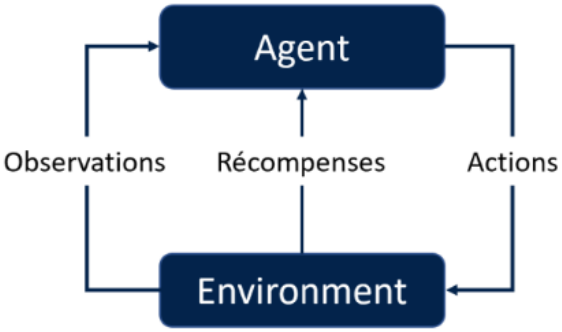

FIGURE 2.1 – Apprentissage par renforcement [12]

L'apprentissage se fait par des tentatives et des erreurs, et un système de récompenses, positif ou négatif, permet à l'agent d'élaborer par lui-même une stratégie définissant quelles actions entreprendre dans une situation donnée afin d'optimiser ses récompenses.

2.3 Apprentissage profond

L'apprentissage profond est largement utilisé dans divers domaines pour son efficacité, principalement grâce aux réseaux neuronaux artificiels (ANNs : Artificial Neural Networks). Ces réseaux, inspirés du fonctionnement du cerveau humain, constituent une méthode moderne et populaire pour manipuler les données. Ils sont composés de nœuds interconnectés qui traitent les informations entrantes et produisent des sorties en utilisant des poids ajustables.
Les ANN sont classés en différentes catégories, y compris les réseaux entièrement connectés (FNNs : Fully-connected Neural Networks), les réseaux convolutionnels (CNN : Convolutional

Neural Networks) et les réseaux récurrents (RNN : Recurrent Neural Networks). Les CNN excellent dans la perception visuelle et l'analyse d'images, tandis que les RNN sont efficaces pour la modélisation des données séquentielles.

2.4 ML et STI

Les données jouent un rôle essentiel dans les STI actuels, provenant de différentes applications couvrant des échelles globales, locales et hybrides [14]. Cette abondance de données offre à l'apprentissage automatique une opportunité naturelle pour extraire des connaissances. Le ML propose des fonctionnalités qui renforcent les STI et constituent les fondements des tâches des applications STI. Dans cette section, nous abordons l'intégration du ML dans les STI et son utilisation dans leurs différentes tâches.

2.4.1 Intégration du ML à l'interieur des STI

L'intégration du ML à l'interieur des STI se compose de trois étapes principales, comme illustré dans la Figure 2.2.

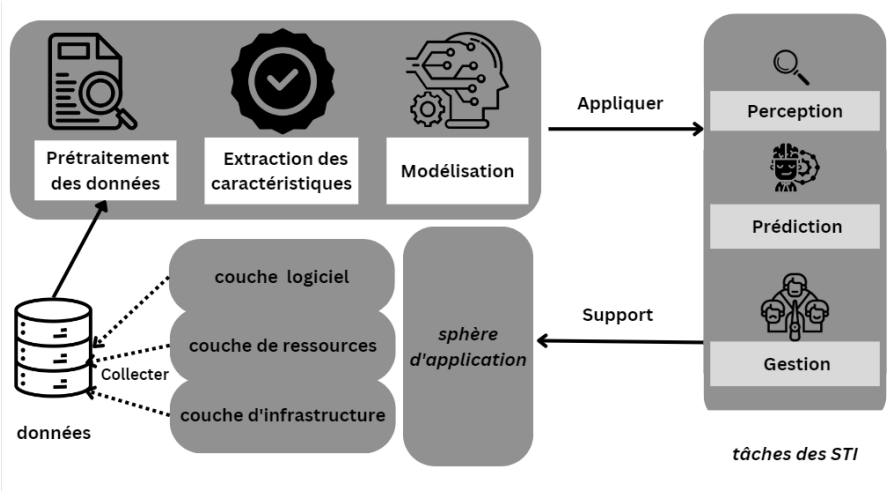

FIGURE 2.2 – Interaction entre ML et STI

— **Prétraitement des données :** Les données brutes sont généralement prétraitées, impliquant des opérations comme le nettoyage et la normalisation des données.

— **Éxtraction des caractéristiques :** Cette étape critique peut être réalisée avec des caractéristiques pré-définies par des experts humains ou via l'apprentissage profond, qui extrait automatiquement des caractéristiques des données.

— **Modélisation :** Le ML est utilisé pour entraîner des modèles pour la régression, la classification, le regroupement et la prise de décision, qui peuvent ensuite être appliqués aux tâches spécifiques des STI.

2.4.2 Éxploitation du ML par les tâches des STI

L'essor des technologies du ML a fortement influencé la transformation des STI et l'amélioration de leurs fonctions clés. De nombreux travaux de recherche ont été entrepris pour repousser les frontières actuelles de ce domaine en constante évolution. Ces travaux sont axés sur la perception, la prédiction et la gestion des STI.

— **Perception visuelle :** La perception visuelle dans les STI bénéficie grandement des méthodes ML. Ces méthodes sont utilisées pour diverses tâches, telles que la détection des panneaux de signalisation [15], la segmentation de l'environnement routier [16], la détection et la classification des véhicules [17], ainsi que la reconnaissance du style de conduite et des piétons [18]. Ces avancées sont cruciales pour améliorer la sécurité et l'efficacité des STI en fournissant une analyse précise des informations visuelles et une gestion efficace des communications.

— **Prédiction :** La prédiction dans les STI englobe diverses tâches telles que la prédiction du flux de trafic, des temps de trajet, du comportement des véhicules et des utilisateurs, ainsi que de l'occupation des routes. Les méthodes traditionnelles de prédiction du trafic utilisent des modèles classiques de séries temporelles, mais les approches basées sur le ML telles que k-NN, SVM, RNN et LSTM (Long Short-Term Memory) offrent de meilleures performances en tenant compte des dépendances temporelles et spatiales. Pour la prédiction du temps de trajet, des approches basées sur le trajet entier ou sur des segments sont utilisées, avec des méthodes ML telles que DBN (Deep Belief Network), RNN, DELM (Deep Extrem Learning Machine)et GAN(Graph Attention Networks). La prédiction du comportement des véhicules et des utilisateurs utilise des CNN pour l'analyse d'images de piétons [19], tandis que les RNN et LSTMs sont utilisés pour prédire leurs actions et trajectoires [20]. Enfin, la prédiction de l'occupation des routes et des places de parking est également une tâche importante dans les STI [21].

— **Gestion :** La gestion ML dans les STI comprend la gestion de l'infrastructure et des ressources. Pour l'infrastructure, la gestion des feux de signalisation vise à atténuer la congestion, avec des méthodes adaptatives comme le Q-learning [22] et les réseaux de neurones pour gérer les signaux en temps réel [23]. La gestion des véhicules implique la planification de trajectoires avec des techniques telles que SVM [24], notamment l'utilisation de CNN pour contrôler les angles de direction à partir d'images brutes. Pour la gestion des ressources, elle répond aux besoins de services exigeants tels que les diffusions vidéo à la demande et les rapports de trafic en direct. Cette gestion efficace des ressources locales et partagées, incluant les unités de service routier et les unités embarquées, est nécessaire. L'apprentissage automatique offre une approche dynamique et efficiente pour gérer les ressources réseau, informatiques, de stockage et énergétiques dans les STI. Les techniques d'apprentissage automatique, telles que le renforcement, le deep learning et les algorithmes de régression, sont utilisées pour optimiser la qualité de service [25], minimiser les coûts de provisionnement [26], étendre la durée de vie des batteries [27] et gérer efficacement les ressources informatiques et réseau.

Le ML ne se limite pas aux tâches des STI, mais trouve également une application importante dans la sécurisation des réseaux véhiculaires. Ces réseaux utilisent des approches basées sur le ML pour résoudre divers problèmes, notamment en raison de la vulnérabilité à différentes attaques dues à la communication sans fil entre les nœuds véhiculaires et/ou l'infrastructure. Ainsi, le ML et ses dérivés sont de plus en plus populaires pour détecter les attaques et traiter divers problèmes de sécurité dans la communication entre véhicules.
Dans la section suivante, nous aborderons plus en détail les différentes techniques du ML utilisées pour détecter et contrer les attaques dans les réseaux véhiculaires.

2.5 ML dans la sécurisation des VANETs

Les réseaux véhiculaires présentent des vulnérabilités face à divers types d'attaques. Pour répondre à ces défis de sécurité, plusieurs solutions cryptographiques ont été proposées dans le passé. Les techniques traditionnelles d'authentification telles que la protection par mot de

passe, l'authentification basée sur des clés et les méthodes biométriques peuvent également être utilisées pour vérifier l'identité des véhicules dans ces réseaux. Cependant, ces méthodes ne permettent pas de distinguer les valeurs transférées entre les véhicules réelles des valeurs falsifiées, et leur mise en œuvre dans des systèmes de sécurité de véhicules à faible puissance reste complexe. Récemment, l'utilisation du ML suscite un intérêt croissant pour renforcer la sécurité des communications entre véhicules, en offrant des prédictions d'attaques plus rapides et plus précises.
Dans cette section, selon des critères d'évaluation nous allons réaliser une étude comparative des travaux de pointe utilisant le ML pour résoudre ces problèmes .

2.5.1 Critères d'évaluation

Les techniques de ML sont évaluées en termes de précision, rappel, score F1 et exactitude.

— **Précision :** mesure la proportion de prédictions correctes parmi toutes les prédictions effectuées par le modèle. Cela permet d'évaluer la capacité du modèle à éviter les erreurs de prédiction. Une précision élevée signifie que le modèle fait peu de prédictions incorrectes par rapport au nombre total de prédictions. Elle est calculée comme suit :

$$Precision = \frac{TP}{TP + FP} * 100$$

— **Rappel (recall en anglais) :** évalue la capacité du modèle à identifier correctement tous les cas positifs parmi ceux qui sont réellement positifs dans un ensemble de données donné. En d'autres termes, il mesure la proportion de vrais positifs détectés par le modèle. Un rappel élevé indique que le modèle parvient à capturer la plupart des cas positifs, tandis qu'un rappel faible suggère qu'il en manque. Il est calculé comme suit :

$$Recall = \frac{TP}{TP + FN} * 100$$

— **score F1 :** est une mesure qui combine la précision et le rappel en un seul chiffre pour évaluer la performance globale du modèle. Il est calculé en prenant la moyenne harmonique de la précision et du rappel, ce qui donne plus de poids aux valeurs les plus basses. Le score F1 est particulièrement utile lorsque les classes dans les données sont déséquilibrées. Un score F1 élevé indique à la fois une bonne précision et un bon rappel, ce qui signifie que le modèle fait peu de fausses prédictions tout en capturant la plupart des cas positifs. Il est calculé comme suit :

$$F1 - Score = \frac{Precision * Recall}{Precision + Recall} * 100$$

— **exactitude (Accuracy en anglais) :** représente la proportion de prédictions correctes sur l'ensemble des prédictions effectuées par le modèle. Elle est calculée en divisant le nombre total de prédictions correctes par le nombre total de prédictions. Elle est calculée comme suit :

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} * 100$$

2.5.2 Travaux antérieur

Koo et al.[28] proposent une méthode pour générer des graphes d'attaque en utilisant le ML en se basant sur une base de données de vulnérabilités, afin de répondre à l'augmentation des menaces de sécurité dans les réseaux. L'approche proposée pour générer un graphe d'attaque se divise en deux phases distinctes. Dans la première phase, les données sont collectées et

traitées, notamment en extrayant des caractéristiques du réseau et des systèmes. Ces données sont ensuite utilisées pour créer une représentation du graphe d'attaque, incluant les relations entre les hôtes, les vulnérabilités spécifiques, et les chemins d'attaque prédits. La seconde phase implique l'utilisation d'algorithmes de ML pour générer le graphe d'attaque. Un modèle de réseau neuronal à trois couches est utilisé pour la classification binaire, avec des caractéristiques locales représentées sous forme de matrices converties en vecteurs ou matrices d'une dimension pour être utilisées comme entrées dans l'algorithme de ML. L'approche propose également une méthode d'apprentissage à plusieurs sorties pour mapper chaque instance d'entrée sur plusieurs sorties, et utilise des algorithmes classiques tels que KNN et les arbres de décision pour résoudre les problèmes de sortie multiples. L'approche proposée montre une précision de détection des chemins d'attaque de 89.52% dans l'approche d'apprentissage à plusieurs sorties et de 80.68% dans l'approche de classification binaire, respectivement.

L'article de Kadam et Sekhar [29], met en lumière l'utilisation du ML dans la détection des attaques par déni de service distribué (DDoS : Distributed Denial-of-Service) dans les réseaux de véhicules autonomes. L'approche proposée utilise des algorithmes de classification tels que SVM et KNN. Les auteurs collectent des données à partir de différentes sources, notamment le Kaggle, puis les entraînent et les testent avec un ratio de 70-30. L'algorithme SVM est utilisé pour classer les activités normales et malveillantes dans la base de données, tandis que l'algorithme KNN est employé pour trouver les voisins les plus proches. La simulation est effectuée sur l'ensemble de données pour détecter les nœuds malveillants ou les attaques. L'approche hybride KSVM proposée pour l'entraînement et la validation des données montre des performances prometteuses, avec des résultats de 92.46% d'exactitude et 95.31% de precision supérieurs à d'autres algorithmes d'apprentissage automatique.

Rashid et al. [30] ont proposé une méthode de détection des nœuds malveillants dans les réseaux de communication véhiculaire en utilisant le ML en temps réel. Ils ont abordé le problème de la détection des attaques DDoS, qui représente l'une des principales menaces dans les environnements VANETs. Pour ce faire, les auteurs ont proposé deux principaux algorithmes : un algorithme d'optimisation utilisé pour ajuster les paramètres du modèle du ML et un algorithme d'optimisation pour le modèle proposé ; cet algorithme prend en compte les spécificités des VANETs pour détecter les nœuds malveillants de manière efficace. Ils ont simulé des scénarios d'attaque avec différentes densités pour évaluer l'efficacité de leur approche. Ils ont utilisé plusieurs modèles d'apprentissage automatique (GBT, LR, MLPC, RF, SVM) pour comparer les résultats. Les résultats de la simulation montrent que l'approche proposée présente une précision, un rappel, un score F1, une spécificité et une sensibilité de 98%, 99%, 98%, 98%, 97% et 96,6% à partir des résultats proposés. Le RF est recommandé pour de meilleurs résultats de détection de comportement malveillant. Les résultats prédisent avec précision que l'architecture proposée gère efficacement la recherche dans l'environnement VANETs.

Yu et al. [31] élaborent la conception d'une plateforme visant à détecter efficacement et à réagir rapidement aux attaques DDoS dans les VANETs reposant sur le réseau défini par logiciel (SDN : Software-Defined Networking). Face aux mécanismes de déclenchement de détection périodique existants, qui sont souvent lents à identifier et à défendre contre les attaques, les auteurs proposent un mécanisme de déclenchement de détection d'anomalies basé sur les messages PACKET_IN dans les réseaux SDN. Lorsqu'un contrôleur SDN reçoit un message PACKET_IN, il peut extraire les données du paquet d'origine, créer un nouvel enregistrement de flux, et remplir les champs de correspondance de l'en-tête dans l'enregistrement. Ensuite, le nombre de paquets (PACKE_COUNT) est défini à 1 dans le flux, et le nombre d'octets (BYTE_COUNT) est défini en fonction de la taille du paquet. Le mécanisme de déclenche-

ment PACKET_IN détecte les messages PACKET_IN à travers leur taux d'anomalie.
Lorsque le contrôleur reçoit une instruction de détection d'attaque, le module de collecte des entrées de table de flux est démarré. Il extrait les informations des entrées de table de flux, notamment les cinq-tuples du réseau (adresses IP source et destination, ports source et destination, et protocole) et le nombre de paquets et d'octets dans l'entrée de flux. Le module extrait également d'autres caractéristiques des flux, telles que le nombre moyen de paquets par flux, le nombre moyen d'octets par flux, le taux d'entrées dans la table de flux, le pourcentage de flux par paire, la vitesse de génération de ports, et l'entropie des caractéristiques de flux.
Le contrôleur intègre un modèle SVM pour détecter les attaques DDoS. Dans la phase de formation, des ensembles de données d'entraînement comprenant des flux de trafic malveillants et normaux sont utilisés pour entraîner le modèle SVM. Dans la phase de détection en temps réel, le module de collecte des entrées de table de flux extrait les caractéristiques des flux et les transmet au modèle SVM pour déterminer s'ils sont malveillants ou normaux. Si un trafic malveillant est détecté, une alerte d'attaque DDoS est générée. Le résultat de la détection était optimal, avec un taux de détection de 97,68 %.

2.6 Théorie des jeux dans les VANETs

La théorie des jeux trouve son origine historique en 1928. En analysant des jeux de société, John von Neumann a rapidement compris la praticabilité de ses approches pour l'analyse des problèmes économiques. Dans son livre "Theory of Games and Economic Behavior", écrit en collaboration avec Oskar Morgenstern en 1944, il applique déjà sa théorie mathématique à des applications économiques. Les concepts de la théorie des jeux fournissent un langage commun pour formuler, structurer, analyser et finalement comprendre différents scénarios stratégiques. En général, la théorie des jeux étudie les situations de conflit, l'interaction entre les agents et leurs décisions. Nous entamerons cette section avec la définition et les éléments constitutifs d'un jeu. Puis nous citerons les types des jeux. Les sous-sections qui suivront seront consacrées aux jeux en forme normale, en forme extensive et aux jeux stocastiques. La dernière sous-section exposera les méthodes de résolution dans la théorie des jeux. Enfin, nous clôturons par les travaux antérieur dans les VANETs.

2.6.1 Définition d'un jeu

Les jeux se caractérisent par un certain nombre de joueurs ou de décideurs qui interagissent, se menacent éventuellement et forment des coalitions, prennent des actions dans des conditions d'incertitude, et reçoivent finalement certains bénéfices ou récompenses, ou éventuellement des punitions ou des pertes.

2.6.2 Éléments constitutifs d'un jeu

Un jeu se compose de plusieurs éléments essentiels qui déterminent sa structure et son déroulement. Voici les principaux composants d'un jeu :

1. **Joueurs**
   L'entité de base de la théorie des jeux est le joueur i. Un joueur peut être interprété comme un individu seul ou un groupe d'individus prenant une décision dans le jeu [32].
2. **Gain**
   Un gain pouvant être un réel (gain numérique), l'étiquette « gagné » / « perdu », ou encore autre chose, et chaque joueur cherche en priorité à maximiser son gain (i.e., à gagner le plus possible, ou à gagner tout court) [33].

3. **Actions**
   Une action désigne toutes les alternatives parmi lesquelles le joueur peut choisir.
4. **Stratégies** : une stratégie est souvent définie comme un plan d'action destiné à accomplir un objectif spécifique d'un joueur. Il existe deux type de stratégies :
   Stratégie pure : une stratégie pure pour le joueur i est un plan d'action déterministe. L'ensemble de toutes les stratégies pures pour le joueur i est désigné par S_i. Un profil de stratégies pures $s = (s_1, s_2, ..., s_n)$, où $s_i \in S_i$ pour tous les $i = 1, 2, ..., n$, décrit une combinaison particulière de stratégies pures choisies par tous les n joueurs dans le jeu [34].
   Stratégie mixte : soit $S_i = \{s_{i1}, s_{i2}, ..., s_{im}\}$ l'ensemble fini de stratégies pures du joueur i. Définissons ΔS_i comme le simplexe de S_i, qui est l'ensemble de toutes les distributions de probabilité sur S_i. Une stratégie mixte pour le joueur i est un élément $\sigma_i \in \Delta S_i$, de telle sorte que $\sigma_i = \{\sigma_i(s_{i1}), \sigma_i(s_{i2}), ..., \sigma_i(s_{im})\}$ est une distribution de probabilité sur S_i, où $\sigma_i(s_i)$ est la probabilité que le joueur i joue s_i [34].
5. **Utilité**
   Chaque joueur reçoit un paiement numérique qui dépend des actions choisies par tous les joueurs. Supposons que le joueur 1 choisit $a_1 \in A_1$, le joueur 2 choisit $a_2 \in A_2$, etc., et le joueur n choisit $a_n \in A_n$. Alors nous notons le paiement au joueur j, pour $j = 1, 2, ..., n$, par $f_j(a_1, a_2, ..., a_n)$, et l'appelons la fonction de paiement pour le joueur j [34].

2.6.3 Types des jeux

Les jeux sont répartis en différentes perspectives et cas, en fonction de leur nature :

1. **Jeu coopératif / non coopératif**
   Le jeu coopératif est un conflit dans lequel les joueurs peuvent interagir les uns avec les autres et se joindre à des groupes pour atteindre le meilleur résultat possible [35]. Un exemple de jeu coopératif est le jeu de cartes bridge, dans lequel les points de chaque joueur sont comptés séparément, mais le couple avec le plus de points gagne. Les jeux sans coopération se divisent en deux types de jeux qui décrivent les situations en détail et donnent des résultats plus précis. Les jeux coopératifs traitent le jeu dans son ensemble. Malgré le fait que ces deux types soient opposés l'un à l'autre, les stratégies peuvent être combinées, ce qui peut apporter plus d'avantages que les deux.
2. **Somme nulle et somme non nulle**
   Un jeu à somme nulle est un jeu dans lequel le gain d'un joueur est égal à la perte d'un autre [35]. Par exemple, un simple désaccord : si vous gagnez le montant N, quelqu'un perd le même montant N. Dans un jeu à somme non nulle, le coût total du jeu peut varier, ce qui peut bénéficier à un joueur sans retirer son prix à un autre. Comme exemple, les échecs sont parfaits ici : en transformant une pièce en une reine, le joueur A augmente le total de ses gains, tout en ne retirant rien au joueur B.
3. **Parallèle et séquentiel**
   Un jeu parallèle est un jeu dans lequel les joueurs effectuent des actions simultanées, ou où l'action d'un joueur est inconnue des autres jusqu'à la fin du cycle général. Dans un jeu séquentiel, chaque joueur dispose d'informations sur le coup précédent de leur adversaire avant de faire leur choix. Et il n'est pas du tout nécessaire que les informations qui mènent au type suivant soient complètes.

2.6.4 Jeux en forme normale

Un jeu en forme normale se compose de :

1. Un nombre fini de joueurs.
2. Un ensemble de stratégies assigné à chaque joueur.
3. Une fonction de gain, qui attribue un certain gain à chaque joueur en fonction de sa stratégie et de la stratégie des autres joueurs.

La fonction de gain attribue à chaque joueur un certain gain en fonction de sa stratégie et de la stratégie des autres joueurs. Si le nombre de joueurs est limité à deux et si leurs ensembles de stratégies ne contiennent que quelques éléments, le résultat de la fonction de gain peut être représenté dans une matrice, la matrice des gains, qui montre les deux joueurs, leurs stratégies et leurs gains.

Exemple

Joueur 1\Joueur 2	L	R
U	1, 3	2, 4
D	1, 0	3, 3

Dans cet exemple, le joueur 1 (vertical) a deux stratégies différentes : Up (U) et Down (D). Le joueur 2 (horizontal) a également deux stratégies différentes, à savoir Left (L) et Right (R).

Les éléments de la matrice sont les résultats pour les deux joueurs lorsqu'ils jouent certaines stratégies, c'est-à-dire que, supposons que le joueur 1 choisisse la stratégie U et le joueur 2 choisisse la stratégie R, le résultat est (2, 4), c'est-à-dire que le gain pour le joueur 1 est de 2 et pour le joueur 2 est de 4.

2.6.5 Jeux en forme extensive

Contrairement aux jeux en forme normale, les règles d'un jeu en forme extensive sont décrites de telle sorte que les agents du jeu exécutent leurs coups consécutivement. Ce jeu est représenté par un arbre de jeu, où chaque nœud représente chaque étape possible du jeu au fur et à mesure qu'il se déroule.

Il existe un nœud unique appelé nœud initial qui représente le début du jeu. Tout nœud qui n'a qu'une seule arête connectée est un nœud terminal et représente la fin du jeu (et aussi un profil de stratégie). Chaque nœud non terminal appartient à un joueur dans le sens où il représente une étape du jeu où c'est au tour de ce joueur de jouer. Chaque arête représente une action possible qu'un joueur peut entreprendre. Chaque nœud terminal a un gain associé à chaque joueur. Ce sont les gains de chaque joueur si la combinaison d'actions requises pour atteindre ce nœud terminal est effectivement jouée.

Exemple

Un joueur en position de monopole, le "titulaire", est confronté à la possible entrée d'un challenger (le challenger peut, par exemple, être une firme désirant entrer sur un marché régit par un monopole ; un homme politique désirant prendre le contrôle d'un parti ; ou un animal souhaitant entrer en compétition pour obtenir le droit de s'accoupler avec un de ses congénères du sexe opposé). Le challenger peut entrer ou non. Si il entre, le "titulaire" peut accepter ou se battre [36]. Les résultats possibles du jeux sont ainsi (Entrer, Accepter), (Entrer, Se battre) ou (Ne pas entrer). Les règles du jeux spécifient par ailleurs que le challenger joue le premier et que le titulaire ne joue que si le challenger a joué "Entrer". Au début d'un jeu séquentiel et après chaque séquence d'événements, un joueur choisit une action. L'ensemble des actions disponibles pour un joueur ne doit pas nécessairement être donné explicitement dans la description du jeu. Il suffit que la description du jeu spécifie les différents résultats et la séquentialité du jeu pour qu'on puisse en déduire l'ensemble des actions disponibles. L'exemple peut donc être représenté comme illustré dans la figure 2.3.

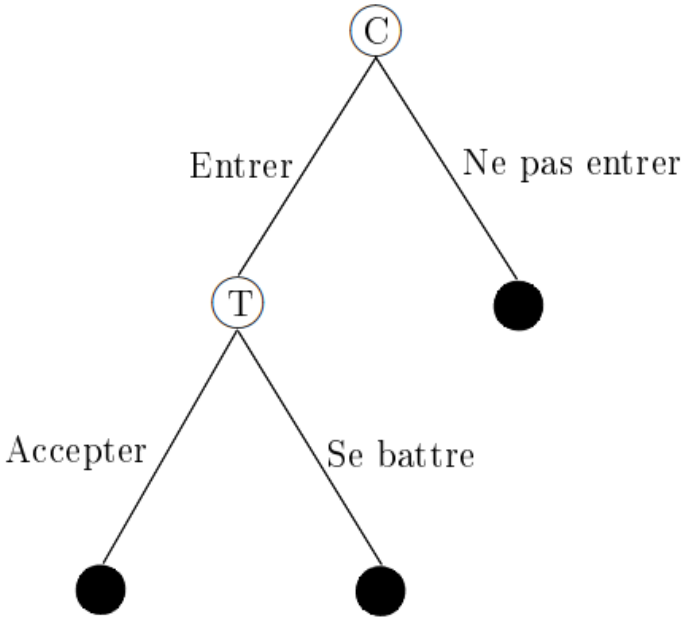

FIGURE 2.3 – Forme extensive de l'exemple [36].

2.6.6 Jeux stochastiques

Un jeu stochastique est un tuple [37] :

$$(n, S, \{A_i\}_{i=1}^{n}, T, \{R_i\}_{i=1}^{n})$$

où :

- n est le nombre d'agents,
- S est un ensemble d'états,
- A_i est l'ensemble des actions disponibles pour l'agent i (et A est l'espace d'actions conjoint $A = A_1 \times A_2 \times \cdots \times A_n$),
- T est une fonction de transition $T : S \times A \times S \to [0, 1]$,
- R_i est une fonction de récompense pour le i-ème agent $R_i : S \times A \to \mathbb{R}$.

Il est important de noter que chaque agent a sa propre fonction de récompense distincte. Nous nous intéressons à déterminer une ligne de conduite pour un agent dans cet environnement. Plus précisément, nous voulons apprendre une politique stationnaire, bien que potentiellement stochastique, $\pi_i : S \to \mathcal{P}(A_i)$, qui associe les états à une distribution de probabilité sur ses actions. L'objectif est de trouver une telle politique qui maximise la récompense future actualisée de l'agent avec un facteur d'actualisation γ.

Chaque état dans un jeu stochastique peut être vu comme un jeu de matrices avec les gains pour chaque action conjointe déterminés par $R_i(s, a)$. Après avoir joué au jeu de matrices et reçu les gains, les joueurs sont transférés à un autre état déterminé par leur action conjointe.

Types de jeux stochastiques

Les jeux purement collaboratifs sont ceux où tous les agents ont la même fonction de récompense. Les jeux purement compétitifs, ou à somme nulle, sont des jeux à deux joueurs où la récompense d'un joueur est toujours le négatif de celle de l'autre. Les jeux stochastiques à somme nulle ont un équilibre de Nash unique.

2.6.7 Méthodes de résolution

Les méthodes et techniques de résolution habituellement employées pour résoudre les jeux en théorie des jeux sont discutées ci-dessous :

Équilibre de Nash

L'équilibre de Nash est une méthode de résolution d'un jeu "non coopératif" impliquant deux ou plusieurs concurrents, dans lequel on suppose que chaque concurrent a connaissance des tactiques d'équilibre ou de stabilité des autres concurrents, et aucun concurrent n'a rien à gagner en variant uniquement sa propre tactique. Si chaque concurrent a choisi une tactique ou une stratégie, et aucun concurrent ne peut en profiter en variant les tactiques tandis que les autres concurrents conservent les leurs, alors les tactiques ou stratégies actuellement choisies et leurs utilités analogues constituent un "équilibre de Nash".

Par exemple, le joueur 1 et le joueur 2 seront en "équilibre de Nash" si le joueur 1 prend la meilleure décision possible, en tenant compte de la décision constante du joueur 2, et que le joueur 2 prend également la meilleure décision possible, en tenant compte de la décision du joueur 1 qui reste constante. De même, un ensemble de concurrents sera en "équilibre de Nash" si chacun prend la meilleure décision possible, en tenant compte des décisions des autres concurrents dans le concours qui restent constantes. Selon Nash, "il existe un équilibre de Nash pour chaque jeu fini".

Elimination de stratégies dominées

La méthode d'elimination de stratégies dominées peut être appliquée à la fois aux jeux avec des stratégies pures et aux jeux avec des stratégies mixtes. Dans un jeu impliquant des stratégies pures, la solution globale est facilement trouvée après l'application de la méthode de domination pour réduire la dimension du problème. Dans un jeu avec des stratégies mixtes, la méthode de domination peut être utilisée pour réduire la dimension du problème avant d'utiliser d'autres méthodes pour résoudre entièrement le problème. Dans un jeu de stratégie mixte donné, la méthode de domination est appliquée comme suit :

1. Si toutes les entrées de la colonne i sont plus grandes ou égales aux entrées équivalentes de n'importe quelle autre colonne j, alors la colonne i est inférieure à la colonne j et est supprimée de la matrice des gains.
2. Si chaque entrée de la ligne i est plus petite ou égale à l'entrée équivalente de n'importe quelle autre ligne j, alors la ligne i est inférieure à la ligne j et est supprimée de la matrice des gains.
3. Répéter (i) et (ii) si une ligne ou une colonne est dominée, sinon arrêter le processus.

La méthode de domination est illustrée avec la matrice des gains suivante :

$$\begin{pmatrix} 1 & 7 & 2 \\ 6 & 2 & 7 \\ 5 & 1 & 6 \end{pmatrix}$$

en supprimant d'abord la ligne 3 qui est dominée par la ligne 2 pour obtenir

$$\begin{pmatrix} 1 & 7 & 2 \\ 6 & 2 & 7 \end{pmatrix}$$

et enfin en supprimant la colonne 3 qui est dominée par la colonne 1 pour obtenir

$$\begin{pmatrix} 1 & 7 \\ 6 & 2 \end{pmatrix}$$

.

Valeur de shapley

Il s'agit d'une technique de résolution ou d'un concept utilisé en théorie des jeux coopératifs. Elle attribue une distribution à tous les joueurs dans un jeu. La distribution est unique et la valeur du jeu dépend de certaines caractéristiques abstraites désirées. En termes simples, la valeur de Shapley attribue des crédits parmi un groupe de joueurs coopérants. Par exemple, supposons qu'il y ait trois joueurs rouges, bleus et verts et que le joueur rouge coopère plus que les joueurs bleus et verts. Maintenant, si l'objectif est de former des paires et ensuite d'attribuer des crédits à celles-ci, chaque paire doit avoir un joueur rouge car il coopère plus que les deux autres. Par conséquent, il peut y avoir deux paires possibles, à savoir joueur rouge-joueur bleu et joueur rouge-joueur vert. Le joueur rouge coopère davantage et, pour cette raison, obtiendra plus de profit que le joueur bleu dans la première paire. De même, le joueur rouge obtiendra plus de profit que le joueur vert dans la deuxième paire.

2.6.8 Travaux antérieur

Dans l'article de Hamza et Bouafia [38], les auteurs proposent une nouvelle approche pour l'analyse des graphes d'attaques basée sur la théorie des jeux afin de réduire les vulnérabilités des réseaux. Cette approche consiste à transformer un problème de sécurité informatique en un jeu à deux joueurs, où chaque joueur cherche à maximiser ses gains tout en minimisant les pertes de l'autre. Ils modélisent la situation comme un jeu à somme non nulle et non coopératif entre un attaquant et un administrateur de réseau. Les auteurs décrivent ensuite les étapes de leur méthode, notamment la représentation du jeu sous forme normale et l'analyse des graphes d'attaques pour réduire les vulnérabilités. Ils appliquent cette approche à un exemple concret de graphe d'attaques pour illustrer son efficacité. En comparant leurs résultats avec une méthode concurrente basée sur la fermeture de ports, ils montrent que leur approche permet de mieux sécuriser les systèmes informatiques en réduisant les vulnérabilités de manière proactive. En conclusion, les auteurs soulignent la simplicité et l'applicabilité de leur approche, tout en identifiant des perspectives pour des recherches futures, notamment l'exploration de jeux répétés non coopératifs.

Hamza et al. [39], proposent une approche (AGA-POSG : Attack Graph Analysis - Partially Observable Stochastic Game) qui est conçue pour répondre aux défis spécifiques de la sécurisation des réseaux IoT, compte tenu de leur complexité et de leur interconnexion. Cette méthode permet de modéliser les interactions entre les différents acteurs du réseau IoT, tels que les périphériques, les utilisateurs et les éventuels attaquants, afin d'éclairer les décisions de l'administrateur en matière de sécurité. La démarche proposée vise à aider l'administrateur à identifier les vulnérabilités menaçant son système et à choisir les meilleures méthodes de correction, en tenant compte de contraintes telles que les coûts. Elle repose sur l'utilisation d'un graphe d'attaques pour représenter les différents chemins qu'un attaquant peut emprunter pour atteindre sa cible, et consiste à analyser ce graphe pour minimiser les pertes et aider à prendre des décisions éclairées pour renforcer la sécurité du réseau. Dans ce contexte, un jeu stochastique partiellement observable, non coopératif à somme nulle et fini est utilisé pour modéliser la situation, où l'administrateur cherche à minimiser les pertes face aux attaques de l'intrus tout en maximisant les gains. Les joueurs, l'attaquant et l'administrateur, ont une connaissance partielle de l'état du jeu, ce qui ajoute une dimension d'incertitude à la prise de décision. La méthodologie présentée propose d'éliminer les stratégies dominées pour déterminer les meilleures stratégies de défense et d'attaque, puis de calculer le coût associé à chaque chemin d'attaque pour résoudre le problème d'analyse des graphes d'attaques. Cette approche

permet d'évaluer les différents scénarios possibles et de prendre des décisions fondées sur des probabilités et des ajustements en fonction des nouvelles informations disponibles. Les avantages de cette approche incluent sa capacité à tenir compte de l'incertitude et de la partialité de l'information, à proposer des stratégies adaptées aux réseaux IoT et à relever les défis de sécurité uniques auxquels ils sont confrontés.

2.7 ML et théorie des jeux dans les VANETs

Asadi [40], présente une approche pour détecter les attaques de botnet sur les objets connectés à l'IoT. Il commence par utiliser la théorie des jeux coopératifs pour sélectionner les caractéristiques les plus importantes dans l'ensemble de données, évaluant l'importance de chaque caractéristique en fonction de son degré de pertinence pour la détection des attaques de botnet IoT. Les scénarios de données incluent des situations normales où les appareils IoT fonctionnent normalement, ainsi que des scénarios de botnet simulant des attaques telles que les sondages, les attaques DoS et les tentatives de vol d'informations. Une fois les données prétraitées, l'auteur utilise la théorie des jeux coopératifs pour sélectionner les dix meilleures caractéristiques parmi un total de 38 dans l'ensemble de données Bot-IoT. Cette sélection est basée sur les critères de succès et la valeur de Shapley de chaque caractéristique, avec pour objectif d'optimiser les métriques d'évaluation telles que l'exactitude, la précision, le rappel, le score F1 et le taux de faux positifs. Pour évaluer l'efficacité de son approche, il compare les performances de différents algorithmes de classification tels que SVM, LSTM et Autoencoder. Les résultats montrent que l'approche proposée offre de meilleures performances dans la détection des attaques de botnet IoT.

Phull et al.[41] fournissent une contribution significative à la gestion des réseaux VANETs en proposant une approche basée sur la théorie des jeux, qui améliore l'efficacité, la fiabilité et la sécurité de la communication dans ces réseaux. Leur méthodologie repose sur une architecture système qui comprend des véhicules sources, des véhicules relais et des points d'accès, et utilise la théorie des jeux pour former des clusters dans le réseau VANET. Concrètement, chaque véhicule est considéré comme un joueur dans un jeu où ils doivent sélectionner les clusters de manière à maximiser certains paramètres sociaux, tels que la fiabilité de la transmission et l'utilisation des ressources de bande passante. Les véhicules sont regroupés en clusters en fonction de ces paramètres, et un chef de cluster est sélectionné à l'aide de l'algorithme K-means. Le système proposé est testé sur diverses caractéristiques, notamment la durée de vie des chefs de cluster, la durée de vie moyenne des membres du cluster, le nombre moyen de réaffiliations, le débit et le taux de perte de paquets. Les résultats indiquent que le VANET s'est très bien comporté avec une haute précision lors de la validation et des tests, avec une précision globale comprise entre 0,97 et 0,99.

2.8 Classification

Les travaux antérieurs dans le domaine des réseaux véhiculaires ont largement exploré l'application de l'apprentissage automatique et la théorie des jeux pour renforcer la sécurité et optimiser les performances. Plusieurs études ont examiné différentes approches et outils pour détecter et répondre aux menaces de sécurité.

Ces recherches mettent en évidence l'importance croissante de l'intégration du ML et de la théorie des jeux dans les systèmes de transport intelligents pour garantir la fiabilité et la sécurité des communications entre les véhicules et les infrastructures routières. Pour mieux comprendre ces approches, une classification selon les techniques utilisées est présentée dans la

Figure 2.4. Malgré cela, il est à noter qu'il existe encore peu de travaux qui exploitent la combinaison de l'apprentissage automatique et de la théorie des jeux spécifiquement dans le contexte des VANETs. Ce constat soulève des opportunités importantes pour développer de nouvelles méthodologies et approches dans ce domaine. Notre proposition de recherche se concentrera ainsi sur l'intégration innovante de la théorie des jeux et de l'apprentissage automatique pour améliorer la sécurité et l'efficacité des réseaux véhiculaires.

FIGURE 2.4 – Diagramme de classification des travaux étudiés

2.9 Conclusion

Ce chapitre a étudié plusieurs travaux existants dans la littérature concernant l'application de l'apprentissage automatique et la théorie des jeux pour renforcer la sécurité et optimiser les performances dans le domaine des réseaux véhiculaires. Les différents modèles présentés ont offert de nouvelles perspectives sur cette problématique, permettant une meilleure compréhension de son importance ainsi que des méthodes pour les aborder et les résoudre. Dans le chapitre suivant, nous exposerons nos propositions.

CHAPITRE

3

PROPOSITIONS

3.1 Introduction

Nous avons vu au sien du chapitre précédent qu'il existe encore peu de recherches qui explorent la théorie des jeux ou la synergie entre l'apprentissage automatique et la théorie des jeux, en particulier dans la sécurisation des reseaux VANETs.

Dans ce contexte, nos propositions, théorie des jeux pour l'analyse des graphes d'attaques et théori des jeux et machine learning pour l'analyse des graphes d'attaques, offrent des approches adaptées à la sécurisation des réseaux VANETs. Ces approches nous ont permis de modéliser les interactions entre les acteurs du réseau VANETs, tels que les véhicules, les ITL (Intelligent Traffic Lights), le RSU (Road Side Unit) et les attaquants potentiels. Elles permetent à l'administrateur de prendre des décisions éclairées pour renforcer la sécurité du réseau VANETs.

3.2 Proposition 1 : Théorie des jeux pour l'analyse des graphes d'attaques

Dans le contexte d'un réseau VANETs, un intrus cherche à compromettre le fonctionnement du reseau, ce qui peut avoir un impact négatif sur la communication entre les véhicules et le RSU, ainsi que sur d'autres fonctionnalités du réseau VANET, et l'administrateur du réseau cherche à réduire les dommages et à augmenter les avantages en réagissant aux attaques lancées par l'attaquant. Dans cette section nous allons proposer une technique qui repose sur l'utilisation d'un graphe d'attaques, qui cartographie tous les chemins potentiels qu'un attaquant pourrait suivre pour compromettre le système. Notre approche consiste à analyser ce graphe d'attaques afin de minimiser les pertes subies par l'administrateur et de lui fournir des informations pour renforcer la sécurité de son réseau.

Notre proposition repose sur l'élimination de stratégies dominées dans la théorie des jeux pour renforcer la sécurité des réseaux VANETs. En utilisant un jeu stochastique partiellement observable, non coopératif à deux joueurs (attaquant et administrateur) à somme nulle et fini.

3.2.1 Cas d'étude

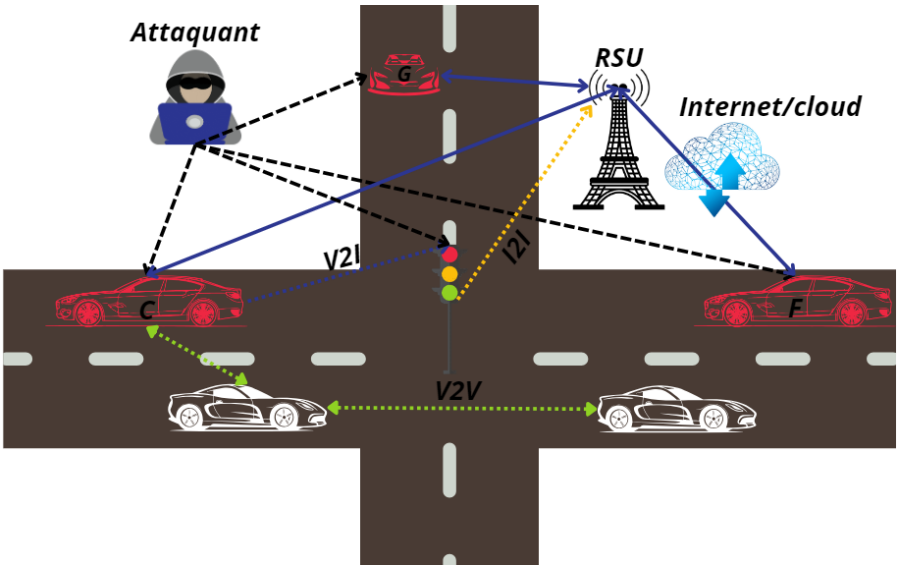

FIGURE 3.1 – exemple d'attaque DDoS dans un reseau VANET.

Une attaque DoS est sévère dans un environnement véhiculaire, mais une attaque DDoS est encore plus grave car le mécanisme de l'attaque est distribué. Dans ce cas, les attaquants

lancent des attaques à partir de différentes localisations. Ils peuvent utiliser différents créneaux horaires pour envoyer les messages. La nature des messages et les créneaux horaires peuvent varier d'un véhicule à l'autre des attaquants. L'objectif est de mettre le réseau hors service. Afin de faciliter la compréhension de notre proposition, nous illustrons notre démarche à travers l'exemple présenté dans la figure 3.1.

En nous appuyant sur la bases de données des vulnérabilités CVE (Common Vulnerabilities and Exposures) [42], nous avons élaboré le tableau 3.1 représentant les vulnérabilités recensées.

Dispositif	**CVE-ID**	**Nom de la Vulnérabilité**	**Niveau de Risque**	**Type de Vulnérabilité**
Véhicule C (V1)	CVE-2024-29454	ROS2 Humble Hawksbill Arbitrary Command execution	Élevé	Exécution de code arbitraire (RCE : Remote Code Execution)
Véhicule F (V2)	CVE-2024-4171	Tenda W30e Stack-based Buffer Overflow Vulnerability	Critique	Débordement de tampon
Véhicule G (V3)	CVE-2023-29389	Toyota RAV4 CAN Bus Message Trust Vulnerability	Critique	Manipulation du bus CAN
Véhicule (V4)	CVE-2024-4904	Byzoro Smart S200 Unrestricted File Upload Vulnerability	Critique	Téléversement de fichiers non restreint
ITL (V5)	CVE-2021-36162	Désérialisation non sécurisée des règles YAML	Élevé	Injection SQL
ITL (V6)	CVE-2024-29018	Docker Networking DNS Forwarding Vulnerability	Élevé	Exfiltration de données
RSU (V7)	CVE-2013-4508	Lighttpd Weak SSL Ciphers Configuration Vulnerability	Moyen	Détournement de flux
RSU (V8)	CVE-2023-47124	Traefik HTTPChallenge Delay exploitation	Moyen	Inondation de requêtes
RSU (V9)	CVE-2023-22397	Fuite de Mémoire DoS de Juniper Networks Junos OS evolved PTX10003	Critique	Épuisement des ressources
RSU (V10)	CVE-2023-20027	Cisco IOS XE IPv4 Virtual Fragmentation Reassembly Denial of Service Vulnerability	Élevé	Envoi de paquets fragmentés

TABLE 3.1 – Exemples de vulnérabilités dans les VANETs

3.2.2 Modélisation

La structure de notre modèle est la suivante :

— **Les joueurs**
L'ensemble des joueurs $N = \{\text{Attaquant, Administrateur}\}$.

— **Les stratégies**
Les stratégies possibles sont représentées par $A = A1 \times A2$, où $A1$ désigne les actions de l'attaquant et $A2$ les actions de l'administrateur.
$A1 = \{$Inondation de requêtes, épuisement des ressources, détournement de flux, manipulation du bus CAN, exécution de code arbitraire, téléversement de fichiers non restreint, SQL injection, exfiltration de données, envoi de paquets fragmentés, débordement de tampon$\}$

1. **Inondation de requêtes :** l'attaquant envoie un grand nombre de requêtes simultanées au RSU, dépassant sa capacité à les traiter efficacement. Cela peut surcharger le RSU et rendre les services indisponibles pour les véhicules légitimes.
2. **Épuisement des ressources :** l'attaquant cible les ressources spécifiques du RSU, telles que la mémoire, le processeur ou la bande passante réseau, en envoyant un volume massif de trafic conçu pour épuiser ces ressources et rendre les services inaccessibles.
3. **Détournement de flux :** l'attaquant coordonne l'envoi de requêtes malveillantes à des intervalles réguliers pour maintenir le RSU continuellement occupé, l'empêchant ainsi de répondre aux demandes légitimes des véhicules.
4. **Manipulation du bus CAN :** est un protocole largement utilisé dans les systèmes embarqués pour la communication entre les différents composants d'un système, tels que les capteurs et les actionneurs dans les véhicules. Une attaque de manipulation du bus CAN consiste à envoyer des messages falsifiés sur le bus CAN pour perturber le fonctionnement du système, modifier les données ou prendre le contrôle des composants du système.
5. **Exécution de code arbitraire :** Cette attaque vise à exploiter les vulnérabilités d'un système pour exécuter du code malveillant sur celui-ci. Cela peut permettre à un attaquant de prendre le contrôle du système et d'accéder à des informations sensibles.
6. **Débordement de tampon :** Cette attaque exploite les vulnérabilités dans la gestion de la mémoire d'un programme pour écrire des données en dehors de la zone prévue de la mémoire tampon. Cela peut entraîner un comportement imprévisible du programme, y compris des plantages, des fuites de données ou l'exécution de code malveillant.
7. **téléversement de fichiers non restreint :** Cette attaque permet à un attaquant de télécharger des fichiers malveillants sur un serveur sans subir de vérifications ou de restrictions appropriées, ce qui peut entraîner la compromission du système et l'exécution de code malveillant.
8. **Injection SQL :** est une vulnérabilité que les attaquants peuvent exploiter pour accéder à des informations confidentielles dans une base de données, modifier des données ou exécuter du code malveillant sur le système cible.
9. **Exfiltration de données :** C'est une attaque où des informations sensibles ou confidentielles sont illégalement transférées depuis un système ou un réseau vers une destination externe contrôlée par l'attaquant.
10. **Envoi de paquets pragmentés :** Cette attaque consiste à envoyer des paquets de données cassés en plusieurs morceaux pour contourner les mesures de sécurité du réseau. Cela peut causer des problèmes, des interruptions de service, ou permettre l'entrée discrète de données malveillantes.

En réponse, les actions de l'administrateur sont :
$A2$ = {Générer une alarme, Blocage d'IP, Isoler l'hôte, Arrêter le processus, Aucune défense}

1. **Générer une alarme :** Cette action peut servir à déclencher une alerte dès que des activités suspectes ou des tentatives d'attaque sont détectées. Cela donne aux défenseurs l'opportunité d'agir rapidement en prenant des mesures préventives pour contrer l'attaque avant qu'elle ne cause des dommages importants.
2. **Blocage d'IP :** Cette mesure vise à interdire l'accès aux adresses IP suspectes ou répertoriées comme étant utilisées par des individus malveillants.
3. **Isoler l'hôte :** Cette mesure implique de séparer les véhicules vulnérables du reste du réseau, les isolant ainsi et empêchant les attaquants d'y accéder depuis le réseau.
4. **Arrêter le processus :** Cette mesure vise à mettre fin aux processus malveillants qui tentent d'exploiter les vulnérabilités.
5. **Aucune défense :** est une stratégie où aucune mesure de sécurité n'est mise en place pour protéger le réseau.

— **Les états**

L'ensemble S représente les états du système à chaque tour de jeu, que ce soit par l'attaquant ou par l'administrateur. Chaque combinaison de stratégies entre les deux joueurs conduit à des états spécifiques :
$S = \{s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7\}$ où :
s_0 : état initial.
s_1 : attaquant réussit à exploiter la vulnérabilité d'exécution de code arbitraire sur le véhicule C.
s_2 : attaquant réussit à exploiter la vulnérabilité débordement de tampon sur le véhicule F.
s_3 : attaquant réussit à exploiter la vulnérabilité de manipulation du bus CAN sur le véhicule G.
s_4 : attaquant réussit à exploiter la vulnérabilité de téléversement de fichiers non restreint sur le véhicule.
s_5 : attaquant réussit à exploiter la vulnérabilité d'injection SQL sur l'ITL.
s_6 : attaquant réussit à exploiter la vulnérabilité d'exfiltration de données sur l'ITL.
s_7 : attaquant réussit à exploiter la vulnérabilité de l'inondation de requêtes sur le RSU.
s_8 : attaquant réussit à exploiter la vulnérabilité de l'épuisement des ressources sur le RSU.
s_9 : attaquant réussit à exploiter la vulnérabilité du détournement de flux sur le RSU.
s_{10} : attaquant réussit à exploiter la vulnérabilité d'envoi de paquets fragmentés sur le RSU.
s_{11} : attaquant réussit à effectuer une attaque DDoS sur le RSU.

— **Fonction de paiement**

Selon le niveau de risque de chaque vulnérabilité nous avons attribuer $\forall s \in S$ les coûts associés aux actions de l'attaquant qui sont représentés dans le tableau 3.2.

(État, Stratégies d'attaque)	Coûts
(s, Inondation de requêtes)	4
(s, Épuisement des ressources)	9
(s, Détournement de flux)	5
(s, Manipulation du bus CAN)	10
(s, Exécution de code arbitraire)	8
(s, Injection SQL)	6
(s, Exfiltration de données)	7
(s, Envoi de paquets fragmentés)	8
(s, Téléversement de fichiers non restreint)	11
(s, Débordement de tampon)	9

TABLE 3.2 – Tableau des coûts de stratégies d'attaques

Les coûts associés aux actions de l'administrateur utilisées contre ces attaques sont représentés dans le tableau 3.3. Ces coûts ont été déterminés en fonction de plusieurs facteurs, notamment la complexité de mise en œuvre, le temps et les ressources nécessaires, ainsi que l'impact potentiel sur le système.

(État, Stratégies de défense)	Coûts
(s, Générer une alarme)	8
(s, Blocage d'IP)	5
(s, Isoler l'hôte)	10
(s, Arrêter le processus)	4
(s, Aucune défense)	0

TABLE 3.3 – Tableau des coûts de stratégies de défense.

Pour déterminer la fonction de paiement R pour chaque joueur, nous appliquons la formule (1) proposée par Anwar et al.[43]

$$R_1(s, a_1, a_2) = \alpha_{1\{I_s(a_1)=\text{accés á un dispositif}\}} - \eta_{1\{I_s(a_1)\neq\text{accés á un dispositif}\}}) - C_1(s, a_1) + C_2(s, a_2) \quad (1)$$

α_1 : représente le coefficient de récompense attribué à l'attaquant lorsque l'accès à un dispositif est réussi, fixé à 0.7 pour notre exemple. Le choix de 0.7 est basé sur le fait qu'une action réussie de l'attaquant est généralement considérée comme plus bénéfique et mérite une récompense élevée.
η_1 : correspond au coefficient de pénalité pour l'attaquant en cas d'échec, fixé à 0.3 pour notre exemple. Le choix de 0.3 reflète le fait que l'échec peut avoir des conséquences mais pas aussi sévères que la récompense pour le succès.
$I_{(\text{condition})}$: agit comme une fonction indicatrice, renvoyant 1 si la condition est vraie et 0 sinon.
a_1 : action de l'attaquant.
a_2 : action de l'administrateur.
$C_1(s, a_1)$: coût associé à l'action de l'attaquant.
$C_2(s, a_2)$: coût associé à l'action de l'administrateur.
d'où R_1 représente la fonction de paiement de l'attaquant et R_2 est la fonction de paiement de l'administrateur. Avec :

$$R_1 = -R_2$$

Dans ce qui suit, nous allons calculer les coûts pour chaque stratégie de l'attaquant et pour chaque réponse possible de l'administrateur :

1. **Inondation de requêtes :**
   R_1(s, Inondation de requêtes, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +4 - 8 = - 4.3
   R_1(s, Inondation de requêtes, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +4 - 5 = - 1.3
   R_1(s, Inondation de requêtes, Isoler l'hôte) = 0.7 * 0 - 0.3 * 1 +4 - 10 = - 6.3
   R_1(s, Inondation de requêtes, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +4 - 4= - 0.3
   R_1(s, Inondation de requêtes, Aucune défense) = 0.7 * 1 - 0.3 * 0 +4 - 0 = 4.7

2. **Épuisement des ressources :**
   R_1(s, Épuisement des ressources, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +9 - 8 = 0.7
   R_1(s, Épuisement des ressources, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +9 - 5 = 3.7
   R_1(s, Épuisement des ressources, Isoler l'hôte) = 0.7 * 0 - 0.3 * 1 +9 - 10 = -1.3
   R_1(s, Épuisement des ressources, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +9 - 4 = 4.7
   R_1(s, Épuisement des ressources, Aucune défense) = 0.7 * 1 - 0.3 * 0 +9 - 0 = 9.7

3. **Détournement de flux :**
   R_1(s, Détournement de flux, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +5 - 8 = - 3.3
   R_1(s, Détournement de flux, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +5 - 5 = -0.3
   R_1(s, Détournement de flux, Isoler l'hôte) = 0.7 * 0 - 0.3 * 1 +5 - 10 = - 5.3
   R_1(s, Détournement de flux, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +5 - 4 = 0.7
   R_1(s, Détournement de flux, Aucune défense) = 0.7 * 1 - 0.3 * 0 +5 - 0 = 5.7

4. **Manipulation du bus CAN :**
   R_1(s, Manipulation du bus CAN, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +10 - 8 = 1.7
   R_1(s, Manipulation du bus CAN, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +10 - 5 = 4.7
   R_1(s, Manipulation du bus CAN, Isoler l'hôte) = 0.7 * 0 - 0.3 * 1 +10 - 10 = -0.3
   R_1(s, Manipulation du bus CAN, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +10 - 4 = 5.7
   R_1(s, Manipulation du bus CAN, Aucune défense) = 0.7 * 1 - 0.3 * 0 +10 - 0 = 10.7

5. **Exécution de code arbitraire :**
   R_1(s, Exécution de code arbitraire, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +8 - 8 = - 0.3
   R_1(s, Exécution de code arbitraire, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +8 - 5 = 2.7
   R_1(s, Exécution de code arbitraire, Isoler l'hôte) = 0.7 * 0 - 0.3 * 1 +8 - 10 = - 2.3
   R_1(s, Exécution de code arbitraire, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +8 - 4 = 3.7
   R_1(s, Exécution de code arbitraire, Aucune défense) = 0.7 * 1 - 0.3 * 0 +8 - 0 = 8.7

6. **Débordement de tampon :**
   R_1(s, Débordement de tampon, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +9 - 8 = 0.7
   R_1(s, Débordement de tampon, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +9 - 5 = 3.7
   R_1(s, Débordement de tampon, Isoler l'hôte) = 0.7 * 0 - 0.3 * 1 +9 - 10 = - 1.3
   R_1(s, Débordement de tampon, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +9 - 4 = 4.7
   R_1(s, Débordement de tampon, Aucune défense) = 0.7 * 1 - 0.3 * 0 +9 - 0 = 9.7
7. **Injection SQL :**
   R_1(s, Injection SQL, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +6 - 8 = -2.3

R_1(s, Injection SQL, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +6 - 5 =0.7
R_1(s, Injection SQL, Isoler l'hôte) = 0.7 * 0 - 0.3 * 1 +6 - 10 = -4.3
R_1(s, Injection SQL, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +6 - 4 = 1.7
R_1(s, Injection SQL, Aucune défense) = 0.7 * 1 - 0.3 * 0 +6 - 0 = 6.7

8. **Exfiltration de données :**
   R_1(s, Exfiltration de données, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +7 - 8 = -1.3
   R_1(s, Exfiltration de données, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +7 - 5 = 1.7
   R_1(s, Exfiltration de données, Isoler l'hôte) = 0.7 * 0 - 0.3 * 1 +7 - 10 = -3.3
   R_1(s, Exfiltration de données, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +7 - 4 = 2.7
   R_1(s, Exfiltration de données, Aucune défense) = 0.7 * 1 - 0.3 * 0 +7 - 0 = 7.7
9. **Envoi de paquets fragmentés :**
   R_1(s, Envoi de paquets fragmentés, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +8 - 8 = -0.3
   R_1(s, Envoi de paquets fragmentés, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +8 - 5 = 2.7
   R_1(s, Envoi de paquets fragmentés, Isoler l'hôte) = 0.7 * 0 - 0.3 * 1 +8 - 10 = -2.3
   R_1(s, Envoi de paquets fragmentés, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +8 - 4 = 3.7
   R_1(s, Envoi de paquets fragmentés, Aucune défense) = 0.7 * 1 - 0.3 * 0 +8 - 0 = 8.7
10. **Téléversement de fichiers non restreint :**
    R_1(s, Téléversement de fichiers non restreint, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +11 - 8 = 2.7
    R_1(s, Téléversement de fichiers non restreint, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +11 - 5 = 5.7
    R_1(s, Téléversement de fichiers non restreint, Isoler l'hôte) = 0.7 * 0 - 0.3 * 1 +11 - 10 = 0.7
    R_1(s, Téléversement de fichiers non restreint, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +11 - 4 = 6.7
    R_1(s, Téléversement de fichiers non restreint, Aucune défense) = 0.7 * 1 - 0.3 * 0 +11 - 0 = 11.7

3.2.3 Forme normale du jeu

Une fois les stratégies de chaque joueur et les coûts associés identifiés, la forme normale du jeu est présentée dans le tableau 3.4.

Attaquant	**Administrateur**				
	Générer une alarme	**Blocage d'IP**	**Isoler l'hôte**	**Arrêter le processus**	**Aucune défense**
Inondation de requête	(-4.3, 4.3)	(-1.3, 1.3)	(-6.3, 6.3)	(-0.3, 0.3)	(4.7, -4.7)
Épuisement des ressources	(0.7, -0.7)	(3.7, -3.7)	(-1.3, 1.3)	(4.7, -4.7)	(9.7, -9.7)
Détournement de flux	(-3.3, 3.3)	(-0.3, 0.3)	(-5.3, 5.3)	(0.7, -0.7)	(5.7, -5.7)
Manipulation du bus CAN	(1.7, -1.7)	(4.7, -4.7)	(-0.3, 0.3)	(5.7, -5.7)	(10.7, -10.7)
Exécution de code arbitraire	(-0.3, 0.3)	(2.7, -2.7)	(-2,3, 2.3)	(3.7, -3.7)	(8.7, -8.7)
Débordement de tampon	(0.7, -0.7)	(3.7, -3.7)	(-1.3, 1.3)	(4.7, -4.7)	(9.7, -9.7)
Injection SQL	(-2.3, 2.3)	(0.7, -0.7)	(-4.3, 4.3)	(1.7, -1.7)	(6.7, -6.7)
Exfiltration de données	(-1.3, 1.3)	(1.7, -1.7)	(-3.3, 3.3)	(2.7, -2.7)	(7.7, -7.7)
Envoi de paquets fragmentés	(-0.3, -0.3)	(2.7, -2.7)	(-2.3, 2.3)	(3.7, -3.7)	(8.7, -8.7)
Téléversement de fichiers non restreint	(2.7, -2.7)	(5.7, -5.7)	(0.7, -0.7)	(6.7, -6.7)	(11.7, -11.7)

TABLE 3.4 – Forme normale du jeu

3.2.4 Analyse du graphe d'attaques

Pour la phase d'analyse, nous appliquons l'algorithme 1 à notre exemple.

Algorithm 1 Algorithme d'élimination itérée des stratégies dominées

1: **Etape 1 :** $A_i^0 = A_i$, initialisation avec A_i.
2: **Etape 2 :** $A_i^1 = \{$stratégies non dominées$\}$, ensemble des stratégies non dominées.
3: **for** $K = 1$ to ∞ **do**
4: **Etape** $K+1$ **:** $A_i^{K+1} = \{s_i \in A_i^K \mid \nexists y_i \in A_i^K, \forall a_{-i}, f_i(y_i, a_{-i}) > f_i(s_i, a_{-i})\}$
5: **end for**
6: **Etape** ∞ **:** $A_i^\infty = \bigcap_k A_i^k$.

3.2.5 Élimination itérée des stratégies dominées

Étape 1 :

$A_1^0 =$ {Inondation de requêtes, épuisement des ressources, détournement de flux, Manipulation du bus CAN, exécution de code arbitraire, téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés, débordement de tampon}

$A_2^0 =$ {Générer une alarme, Blocage d'IP, Isoler l'hôte, Arrêter le processus, Aucune défense}

— **Aucune défense** : Comparons avec "Arrêter le processus" :

— Inondation de requêtes : $-4.7 < 0.3$

- Épuisement des ressources : $-9.7 < -4.7$
- Détournement de flux : $-5.7 < -0.7$
- Manipulation du bus CAN : $-10.7 < -5.7$
- Exécution de code arbitraire : $-8.7 < -3.7$
- Débordement de tampon : $-9.7 < -4.7$
- SQL injection : $-6.7 < -1.7$
- Exfiltration de données : $-7.7 < -2.7$
- Envoi de paquets fragmentés : $-8.7 < -3.7$
- Téléversement de fichiers non restreint : $-11.7 < -6.7$

"Arrêter le processus" donne de meilleurs résultats dans chaque cas. Éliminer "Aucune défense".

Après cette élimination, les stratégies de l'administrateur restantes sont :

- Générer une alarme
- Blocage d'IP
- Isoler l'hôte
- Arrêter le processus

Étape 2 :

A_1^1 = {Inondation de requêtes, épuisement des ressources, détournement de flux, Manipulation du bus CAN, exécution de code arbitraire, téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés, débordement de tampon}

A_2^1 = {Générer une alarme, Blocage d'IP, Isoler l'hôte, Arrêter le processus}

Nous comparons chaque stratégie de l'attaquant avec les stratégies restantes de l'administrateur :

- **Inondation de requête** : Comparons avec "Téléversement de fichiers non restreint" :
  - Générer une alarme : $-4.3 < 2.7$
  - Blocage d'IP : $-1.3 < 5.7$
  - Isoler l'hôte : $-6.3 < 0.7$
  - Arrêter le processus : $-0.3 < 6.7$

  "Téléversement de fichiers non restreint" donne de meilleurs résultats dans chaque cas. Éliminer "Inondation de requête".

Après cette élimination, les stratégies de l'attaquant restantes sont :

- Épuisement des ressources
- Détournement de flux
- Manipulation du bus CAN
- Exécution de code arbitraire
- Débordement de tampon
- Injection SQL
- Exfiltration de données
- Envoi de paquets fragmentés
- Téléversement de fichiers non restreint

Étape 3 :

A_1^2 = { Épuisement des ressources, détournement de flux, Manipulation du bus CAN, exécution de code arbitraire, téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés, débordement de tampon}

A_2^2 = {Générer une alarme, Blocage d'IP, Isoler l'hôte, Arrêter le processus}

— **Arrêter le processus** : Comparons avec "Isoler l'hôte" :
  — Épuisement des ressources : $-4.7 < 1.3$
  — Détournement de flux : $-0.7 < 5.3$
  — Manipulation du bus CAN : $-5.7 < 0.3$
  — Exécution de code arbitraire : $-3.7 < 2.3$
  — Débordement de tampon : $-4.7 < 1.3$
  — SQL injection : $-1.7 < 4.3$
  — Exfiltration de données : $-2.7 < 3.3$
  — Envoi de paquets fragmentés : $-3.7 < 2.3$
  — Téléversement de fichiers non restreint : $-6.7 < -0.7$

  "Isoler l'hôte" donne de meilleurs résultats dans chaque cas. Éliminer "Arrêter le processus".

Après cette élimination, les stratégies de l'administrateur restantes sont :

— Générer une alarme

— Blocage d'IP

— Isoler l'hôte

Étape 4 :

A_1^4 = { Épuisement des ressources, détournement de flux, Manipulation du bus CAN, exécution de code arbitraire, téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés, débordement de tampon}

A_2^3 = {Générer une alarme, Blocage d'IP, Isoler l'hôte}

— **Téléversement de fichiers non restreint** : Comparons avec "Épuisement des ressources" :
  — Générer une alarme : $o.7 < 2.7$
  — Blocage d'IP : $3.7 < 5.7$
  — Isoler l'hôte : $-1.3 < 0.7$

  "Téléversement de fichiers non restreint" donne de meilleurs résultats dans chaque cas. Éliminer "Epuisement des ressources".

Après cette élimination, les stratégies de l'attaquant restantes sont :

— Détournement de flux

— Manipulation du bus CAN

— Exécution de code arbitraire

— Débordement de tampon

— Injection SQL

— Exfiltration de données

— Envoi de paquets fragmentés

— Téléversement de fichiers non restreint

Étape 5 :

A_1^4 = { Détournement de flux, Manipulation du bus CAN, exécution de code arbitraire, téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés, débordement de tampon}

A_2^4 = {Générer une alarme, Blocage d'IP, Isoler l'hôte, Arrêter le processus}

— **Blocage d'IP** : Comparons avec "Isoler l'hôte" :

- Détournement de flux : $0.3 < 5.3$
- Manipulation du bus CAN : $-4.7 < 0.3$
- Exécution de code arbitraire : $-2.7 < 2.3$
- Débordement de tampon : $-3.7 < 1.3$
- SQL injection : $-0.7 < 4.3$
- Exfiltration de données : $-1.7 < 3.3$
- Envoi de paquets fragmentés : $-2.7 < 2.3$
- Téléversement de fichiers non restreint : $-5.7 < -0.7$

"Isoler l'hôte" donne de meilleurs résultats dans chaque cas. Éliminer "Blocage d'IP".

Après cette élimination, les stratégies de l'administrateur restantes sont :

- Générer une alarme
- Isoler l'hôte

Étape 6 :

A_1^5 = {Détournement de flux, Manipulation du bus CAN, exécution de code arbitraire, téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés, débordement de tampon}

A_2^5 = {Générer une alarme, Isoler l'hôte}

- **Téléversement de fichiers non restreint** : Comparons avec "Détournement de flux" :
  - Générer une alarme : $-3.3 < 2.7$
  - Isoler l'hôte : $-5.3 < 0.7$

  "Téléversement de fichiers non restreint" donne de meilleurs résultats dans chaque cas. Éliminer "Détournement de flux".

Après cette élimination, les stratégies de l'attaquant restantes sont :

- Manipulation du bus CAN
- Exécution de code arbitraire
- Débordement de tampon
- Injection SQL
- Exfiltration de données
- Envoi de paquets fragmentés
- Téléversement de fichiers non restreint

Étape 7 :

A_1^4 = { Manipulation du bus CAN, exécution de code arbitraire, téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés, débordement de tampon}

A_2^4 = {Générer une alarme, Isoler l'hôte}

- **Générer une alarme** : Comparons avec "Isoler l'hôte" :
  - Manipulation du bus CAN : $-1.7 < 0.3$
  - Exécution de code arbitraire : $0.3 < 2.3$
  - Débordement de tampon : $-0.7 < 1.3$
  - SQL injection : $2.3 < 4.3$
  - Exfiltration de données : $1.3 < 3.3$
  - Envoi de paquets fragmentés : $-0.3 < 2.3$
  - Téléversement de fichiers non restreint : $-2.7 < -0.7$

"Isoler l'hôte" donne de meilleurs résultats dans chaque cas. Éliminer "Générer une alarme".

Après cette élimination, les stratégies de l'administrateur restantes sont :

— Isoler l'hôte

Étape 8 :
A_1^5 = { Manipulation du bus CAN, exécution de code arbitraire, téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés, débordement de tampon}
A_2^5 = {Isoler l'hôte}

— **Téléversement de fichiers non restreint** : Comparons avec "Manipulation de bus CAN" :

  — Isoler l'hôte : $-0.3 < 0.7$

  "Téléversement de fichiers non restreint" donne de meilleurs résultats dans chaque cas. Éliminer "Manipulation de bus CAN".

Après cette élimination, les stratégies de l'attaquant restantes sont :

— Exécution de code arbitraire

— Débordement de tampon

— Injection SQL

— Exfiltration de données

— Envoi de paquets fragmentés

— Téléversement de fichiers non restreint

Étape 9 :
A_1^5 = {Exécution de code arbitraire, téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés, débordement de tampon}
A_2^5 = {Isoler l'hôte}

— **Téléversement de fichiers non restreint** : Comparons avec "Exécution de code arbitraire" :

  — Isoler l'hôte : $-2.3 < 0.7$

  "Téléversement de fichiers non restreint" donne de meilleurs résultats dans chaque cas. Éliminer "Exécution de code arbitraire".

Après cette élimination, les stratégies de l'attaquant restantes sont :

— Débordement de tampon

— Injection SQL

— Exfiltration de données

— Envoi de paquets fragmentés

— Téléversement de fichiers non restreint

Étape 9 :
A_1^5 = { téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés, débordement de tampon}
A_2^5 = {Isoler l'hôte}

— **Téléversement de fichiers non restreint** : Comparons avec "débordement de tampon" :

— Isoler l'hôte : $-1.3 < 0.7$

"Téléversement de fichiers non restreint" donne de meilleurs résultats dans chaque cas. Éliminer "débordement de tampon".

Après cette élimination, les stratégies de l'attaquant restantes sont :

— Injection SQL

— Exfiltration de données

— Envoi de paquets fragmentés

— Téléversement de fichiers non restreint

Étape 10 :

A_1^5 = { téléversement de fichiers non restreint, SQL injection, exfiltration de données, Envoi de paquets fragmentés}

A_2^5 = {Isoler l'hôte}

— **Téléversement de fichiers non restreint** : Comparons avec " SQL injection" :

— Isoler l'hôte : $-4.3 < 0.7$

"Téléversement de fichiers non restreint" donne de meilleurs résultats dans chaque cas. Éliminer " SQL injection".

Après cette élimination, les stratégies de l'attaquant restantes sont :

— Exfiltration de données

— Envoi de paquets fragmentés

— Téléversement de fichiers non restreint

Étape 11 :

A_1^5 = { téléversement de fichiers non restreint, exfiltration de données, Envoi de paquets fragmentés}

A_2^5 = {Isoler l'hôte}

— **Téléversement de fichiers non restreint** : Comparons avec " SQL injection" :

— Isoler l'hôte : $-3.3 < 0.7$

"Téléversement de fichiers non restreint" donne de meilleurs résultats dans chaque cas. Éliminer " exfiltration de données".

Après cette élimination, les stratégies de l'attaquant restantes sont :

— Envoi de paquets fragmentés

— Téléversement de fichiers non restreint

Étape 12 :

A_1^5 = { téléversement de fichiers non restreint, Envoi de paquets fragmentés}

A_2^5 = {Isoler l'hôte}

— **Téléversement de fichiers non restreint** : Comparons avec " Envoi de paquets fragmentés" :

— Isoler l'hôte : $-2.3 < 0.7$

"Téléversement de fichiers non restreint" donne de meilleurs résultats dans chaque cas. Éliminer " Envoi de paquets fragmentés".

Après cette élimination, les stratégies de l'attaquant restantes sont :

— Téléversement de fichiers non restreint

Ainsi, nous conclurons que le jeu a un équilibre de Nash qui est le profile de stratégies (Téléversement de fichiers non restreint, Isoler l'hôte) = (0.7, -0.7)

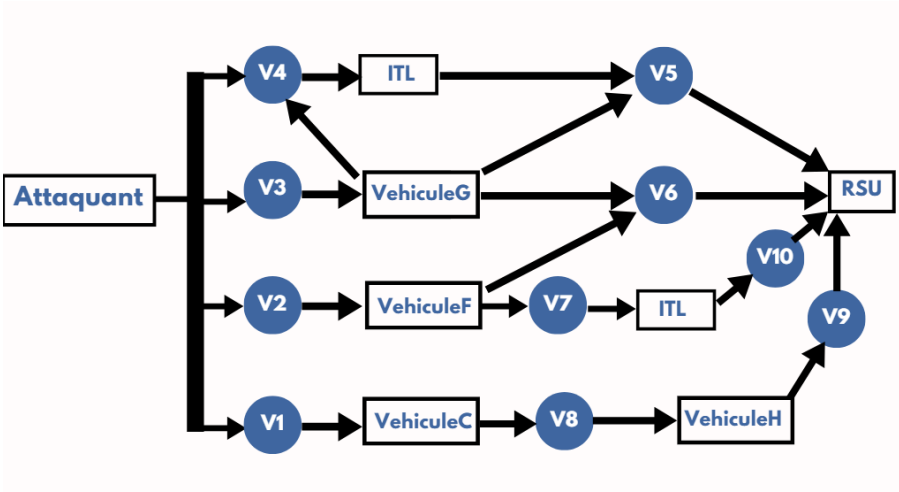

FIGURE 3.2 – Graphe d'attaque.

3.2.6 Coût des chemins

La figure 3.2 représente le graphe d'attaques que nous avons construit et qui va être analysé. Les rectangles représentent les dispositifs de la topologie représentée par la Figure 3.1 et les cercles représentent les vulnérabilités associer à chaque dispositif et que l'intrus utilise comme stratégies d'attaques. En appliquant le processus d'élimination des stratégies dominées sur le graphe, certains dispositifs, comme l'ITL, seront retirés. Pour prendre des décisions éclairées, il est crucial d'évaluer les coûts liés à la suppression d'un chemin spécifique dans ce cadre.

Lorsque nous appliquons l'étape d'élimination des stratégies dans un graphe d'attaques, nous cherchons à identifier tous les chemins qui contiennent des vulnérabilités qui ont été éliminées. L'objectif est ensuite de calculer le coût associé á chaque chemin, afin de déterminer quel chemin peut être supprimé ceci correspond à celui qui a un coût minimum.
Les chemins d'attaques sont :

Chemin 1 = {V5, RSU}
Chemin 2 = {V3, VehiculeG, V5, RSU}
Chemin 3 = {V6, RSU}
Chemin 4 = {V3, VehiculeG, V6, RSU}
Chemin 5 = {V2, VehiculeF, V6, RSU}
Chemin 6 = {V10, RSU}
Chemin 7 = {V2, VehiculeF, V7, ITL, V10, RSU}
chemin 8 = {V9, RSU}
chemin 9 = {V1, VehiculeC, V8, VehiculeH, V9, RSU}

On va commencer par l'identification du risque de sécurité, noté R.
R est calculé par la formule (2) définie par Feng chen et al. [44] :

$$R = \frac{1}{K} \times W + (1 - W) \sum_{i=1}^{m} \frac{1}{l_i} \tag{2}$$

Où :

m : nombre de chemins d'attaques.
l_i : distance du chemin d'attaque i.
K : différentes vulnérabilités composant les chemins d'attaques.

W : probabilité de résistance de la connaissance de l'attaquant.
On a :

$l_1 = 1$, $l_2 = 2$, $l_3 = 1$, $l_4 = 2$, $l_5 = 2$, $l_6 = 1$, $l_7 = 3$, $l_8 = 1$, $l_9 = 3$, $m = 9$, $k = 9$, w= 0.5 donc :

$$R = \frac{1}{9} \times 0.5 + (1 - 0.5) \times \left(\frac{1}{1} + \frac{1}{2} + \frac{1}{1} + \frac{1}{2} + \frac{1}{2} + \frac{1}{1} + \frac{1}{3} + \frac{1}{1} + \frac{1}{3}\right)$$
$$R = 0.055 + 0.5 \times 6,16$$
$$R = 0.055 + 3.08$$
$$R = 3.13$$

Le tableau 3.5 présente le degré de chaque vulnérabilité où le nombre de vulnérabilité égale à 9 :

Les différentes vulnérabilités	**Nombre d'arcs liés**
Éxécution de code arbitraire (V1)	1
Débordement de tempon (V2)	1
Manipulation de bus CAN (V3)	1
Injection SQL (V5)	2
Exfiltration de données (V6)	2
Détournement de flux (V7)	1
Inondation de requêtes (V8)	1
Épuisement des ressources (V9)	1
Envoi de paquets fragmentés (V10)	1

TABLE 3.5 – Tableau des degrés de chaque vulnérabilité.

Pour calculer le coût de chaque chemin, nous définissons la formule (3) :

$$\text{Cout}(chemin_i) = \sum_{j=1}^{n} D \times R \times \frac{1}{Nb} \tag{3}$$

Où :

n : nombre de vulnérabilités composant un chemin donné.
D : somme des degrés de vulnérabilités composant les chemins.
R : risque de sécurité.
N_b : nombre des différentes vulnérabilités présentes dans les chemin.
En appliquant la formule on trouve :

$$\text{Chemin1} = \frac{(2) \times 3.13}{9} = 0.6956$$

$$\text{Chemin2} = \frac{(1 + 2) \times 3.13}{9} = 1.0433$$

$$\text{Chemin3} = \frac{(2) \times 3.13}{9} = 0.6956$$

$$\text{Chemin4} = \frac{(1 + 2) \times 3.13}{9} = 1.0433$$

$$\text{Chemin5} = \frac{(1+2) \times 3.13}{9} = 1.0433$$

$$\text{Chemin6} = \frac{(1) \times 3.13}{9} = 0.3478$$

$$\text{Chemin7} = \frac{(1+1+1) \times 3.13}{9} = 1.0433$$

$$\text{Chemin8} = \frac{(1) \times 3.13}{9} = 0.3478$$

$$\text{Chemin9} = \frac{(1+1+1) \times 3.13}{9} = 1.0433$$

Parmi les différents chemins à traiter, les plus appropriés dans cet exemple sont le Chemin6 et le Chemin8. Supposons que nous décidions de supprimer le chemin6. En conséquence, la vulnérabilité V10 sera éliminée, et le graphe mis à jour est illustré dans la Figure 3.3.

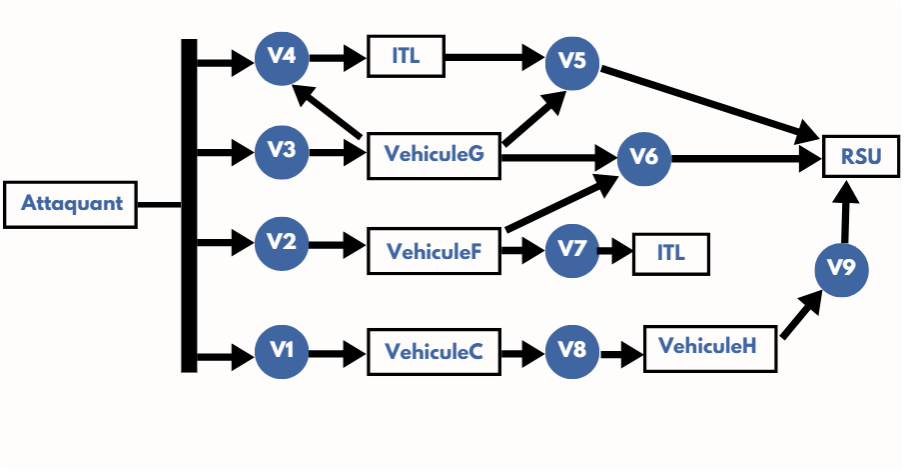

FIGURE 3.3 – Graphe d'attaque aprés la suppression de V10.

Si nous décidions de supprimer le chemin8. En conséquence, la vulnérabilité V9 sera éliminée, et le graphe mis à jour est illustré dans la Figure 3.4.

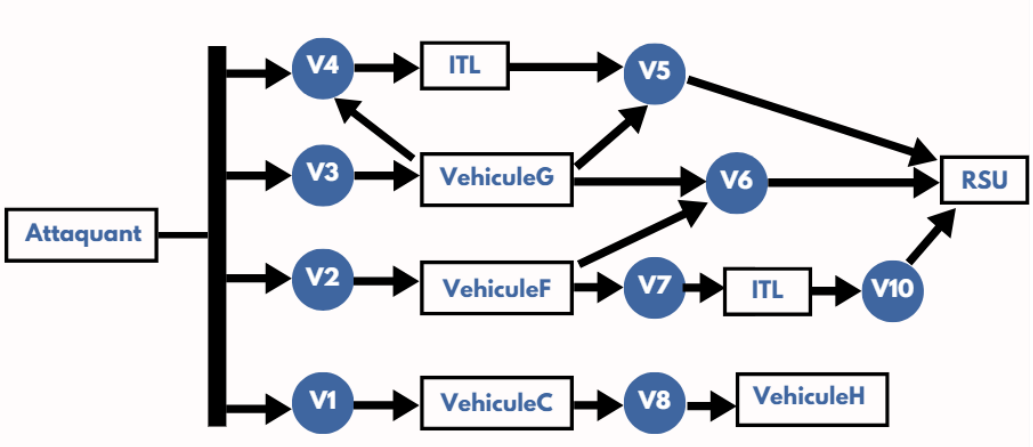

FIGURE 3.4 – Graphe d'attaque aprés la suppression de V9.

3.2.7 Evaluation

Après avoir analysé le graphe d'attaques et éliminé les vulnérabilités des chemins 6 et 8, nous sommes en mesure de prendre une décision éclairée sur les vulnérabilités restantes à traiter. En consultant le tableau 3.1, nous observons que la vulnérabilité V9 présente un niveau de risque critique, tandis que la vulnérabilité V10 présente un risque élevé. Il est donc préférable de supprimer la vulnirabilité V9.
pour la vulnirabilité V10, une mesure de sécurité efficace serait de mettre en œuvre des mécanismes de filtrage des entrées et des sorties pour détecter et bloquer les tentatives d'exploitation de cette faille. De plus, il est crucial de maintenir le système à jour avec les correctifs de sécurité fournis par le fournisseur, et de surveiller activement les comportements suspects du système qui pourraient indiquer une exploitation de la vulnérabilité. Enfin, une stratégie de limitation des accès et des privilèges peut également réduire le risque d'exposition à cette faille.

3.3 Proposition 2 : Théorie des jeux et Machine learning pour l'analyse des graphes d'attaques

Dans cette section nous avons développée une technique qui repose sur l'utilisation d'un graphe d'attaques, qui cartographie tous les chemins potentiels qu'un attaquant pourrait suivre pour compromettre le système. Notre approche consiste à analyser ce graphe d'attaques afin de minimiser les pertes subies par l'administrateur et de lui fournir des informations pour renforcer la sécurité de son réseau.

Notre proposition combine la théorie des jeux et l'apprentissage automatique pour renforcer la sécurité des réseaux VANETs. En utilisant un jeu stochastique partiellement observable, non coopératif à deux joueurs (attaquant et administrateur) à somme nulle et fini et l'algorithme K-means, nous identifions les stratégies de défense les plus efficaces en concentrant notre attention sur les groupes de stratégies les plus coûteux à contrer.

3.3.1 Cas d'étude

Afin de faciliter la compréhension de notre proposition, nous illustrons notre démarche à travers l'exemple présenté dans la Figure 3.5 qui décrit une attaque DDoS contre le RSU où un groupe de trois véhicules infectés (C, F, G) lance une attaque DDoS à partir de différentes localisations sur le RSU.

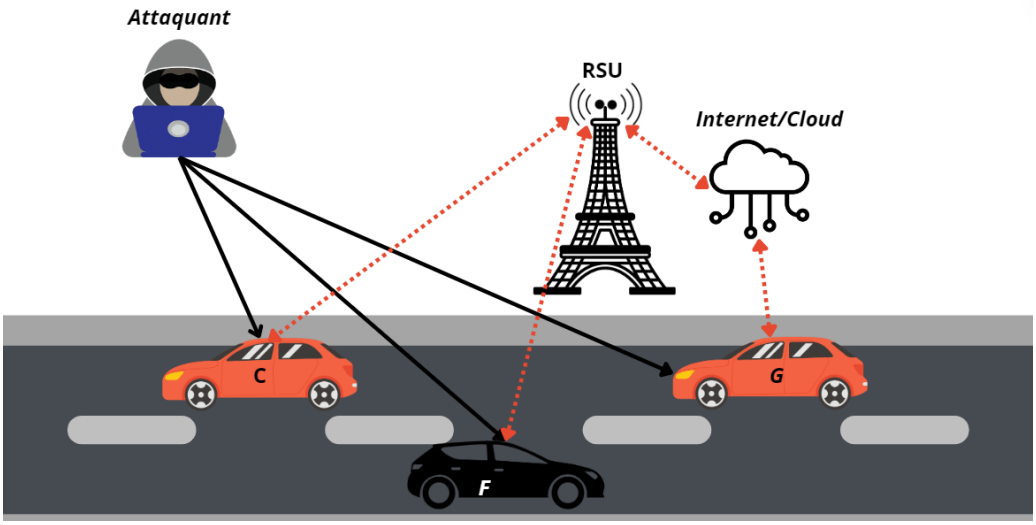

FIGURE 3.5 – exemple d'attaque DDoS dans la communication véhicule RSU.

En nous appuyant sur la base de données des vulnérabilités CVE [42], nous avons élaboré le tableau 3.6 représentant les vulnérabilités recensées.

3.3.2 Modélisation

La structure de notre modèle est la suivante :

— **Les joueurs**
L'ensemble des joueurs N = {Attaquant, Administrateur}.

— **Les stratégies**
Les stratégies possibles sont représentées par $A = A1 \times A2$, où $A1$ désigne les actions de l'attaquant et $A2$ les actions de l'administrateur.
$A1$ = {Inondation de requêtes, épuisement des ressources, détournement de flux, Manipulation du bus CAN, exécution de code arbitraire, débordement de tampon}
En réponse, les actions de l'administrateur sont :
$A2$ = {Générer une alarme, Blocage d'IP, Isoler le véhicule, Arrêter le processus, Aucune défense}

— **Les états**
L'ensemble S représente les états du système à chaque tour de jeu, que ce soit par l'attaquant ou par l'administrateur. Chaque combinaison de stratégies entre les deux joueurs conduit à des états spécifiques :
$S = \{s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7\}$ où :
s_0 : état initial.
s_1 : attaquant réussit à exploiter la vulnérabilité d'exécution de code arbitraire sur le véhicule C.

Dispositif	**CVe-ID**	**Nom de la Vulnérabilité**	**Niveau de Risque**	**Type de Vulnérabilité**
Véhicule C (V1)	CVe-2024-29454	ROS2 Humble Hawksbill Arbitrary Command execution	élevé	exécution de code arbitraire (RCE : Remote Code Execution)
Véhicule F (V2)	CVe-2024-4171	Tenda W30e Stack-based Buffer Overflow Vulnerability	critique	Débordement de tampon
Véhicule G (V3)	CVe-2023-29389	Toyota RAV4 CAN Bus Message Trust Vulnerability	critique	Manipulation du bus CAN
RSU (V4)	CVe-2013-4508	lighttpd Weak SSL Ciphers Configuration Vulnerability	moyen	Détournement de flux.
RSU (V5)	CVe-2023-47124	Traefik HTTP-Challenge Delay exploitation	moyen	Inondation de requêtes
RSU (V6)	CVe-2023-22397	Fuite de Mémoire DoS de Juniper Networks Junos OS evolved PTX10003	Critique	Épuisement des ressources

TABLE 3.6 – Exemples de vulnérabilités(Véhicule/RSU)

s_2 : attaquant réussit à exploiter la vulnérabilité Débordement de tampon sur le véhicule F.
s_3 : attaquant réussit à exploiter la vulnérabilité de manipulation du bus CAN sur le véhicule G.
s_4 : attaquant réussit à exploiter la vulnérabilité de l'inondation de requêtes sur le RSU.
s_5 : attaquant réussit à exploiter la vulnérabilité de l'épuisement des ressources sur le RSU.
s_6 : attaquant réussit à exploiter la vulnérabilité du détournement de flux sur le RSU.
s_7 : attaquant réussit à effectuer une attaque DDoS sur le RSU.

— **Fonction de paiement**
Selon le niveau de risque de chaque vulnérabilité nous avons attribuer $\forall s \in S$ les coûts associés aux actions de l'attaquant qui sont représentés dans le tableau 3.7.

(État, Stratégies d'attaque)	Coûts
(s, Inondation de requêtes)	4
(s, Épuisement des ressources)	9
(s, Détournement de flux)	6
(s, Manipulation du bus CAN)	10
(s, Exécution de code arbitraire)	8
(s, Débordement de tampon)	9

TABLE 3.7 – Tableau des coûts de stratégies d'attaques

Les coûts associés aux actions de l'administrateur utilisées contre ces attaques sont représentés dans le tableau 3.8.

(État, Stratégies de défense)	Coûts
(s, Générer une alarme)	8
(s, Blocage d'IP)	5
(s, Isoler le véhicule)	10
(s, Arrêter le processus)	4
(s, Aucune défense)	0

TABLE 3.8 – Tableau des coûts de stratégies de défense.

Pour déterminer la fonction de paiement R pour chaque joueur, nous appliquons la formule (1).
Dans ce qui suit, nous allons calculer les coûts pour chaque stratégie de l'attaquant pour chaque réponse possible de l'administrateur :

1. **Inondation de requêtes :**
   R_1(s, Inondation de requêtes, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +4 - 8 = - 4.3
   R_1(s, Inondation de requêtes, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +4 - 5 = - 1.3
   R_1(s, Inondation de requêtes, Isoler le véhicule) = 0.7 * 0 - 0.3 * 1 +4 - 10 = - 6.3
   R_1(s, Inondation de requêtes, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +4 - 4= - 0.3
   R_1(s, Inondation de requêtes, Aucune défense) = 0.7 * 1 - 0.3 * 0 +4 - 0 = 4.7

2. **Épuisement des ressources :**
   R_1(s, Épuisement des ressources, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +9 - 8 = 0.7
   R_1(s, Épuisement des ressources, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +9 - 5 = 3.7
   R_1(s, Épuisement des ressources, Isoler le véhicule) = 0.7 * 0 - 0.3 * 1 +9 - 10 = -1.3
   R_1(s, Épuisement des ressources, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +9 - 4 = 4.7
   R_1(s, Épuisement des ressources, Aucune défense) = 0.7 * 1 - 0.3 * 0 +9 - 0 = 9.7

3. **Détournement de flux :**
   R_1(s, Détournement de flux, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +6 - 8 = - 2.3
   R_1(s, Détournement de flux, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +6 - 5 = 0.7

R_1(s, Détournement de flux, Isoler le véhicule) = 0.7 * 0 - 0.3 * 1 +6 - 10 = - 4.3
R_1(s, Détournement de flux, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +6 - 4 = 1.7
R_1(s, Détournement de flux, Aucune défense) = 0.7 * 1 - 0.3 * 0 +6 - 0 = 6.7

4. **Manipulation du bus CAN :**
   R_1(s, Manipulation du bus CAN, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +10 - 8 = 1.7
   R_1(s, Manipulation du bus CAN, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +10 - 5 = 4.7
   R_1(s, Manipulation du bus CAN, Isoler le véhicule) = 0.7 * 0 - 0.3 * 1 +10 - 10 = -0.3
   R_1(s, Manipulation du bus CAN, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +10 - 4 = 5.7
   R_1(s, Manipulation du bus CAN, Aucune défense) = 0.7 * 1 - 0.3 * 0 +10 - 0 = 10.7

5. **Exécution de code arbitraire :**
   R_1(s, Exécution de code arbitraire, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +8 - 8 = - 0.3
   R_1(s, Exécution de code arbitraire, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +8 - 5 = 2.7
   R_1(s, Exécution de code arbitraire, Isoler le véhicule) = 0.7 * 0 - 0.3 * 1 +8 - 10 = - 2.3
   R_1(s, Exécution de code arbitraire, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +8 - 4 = 3.7
   R_1(s, Exécution de code arbitraire, Aucune défense) = 0.7 * 1 - 0.3 * 0 +8 - 0 = 8.7

6. **Débordement de tampon :**
   R_1(s, Débordement de tampon, Générer une alarme) = 0.7 * 0 - 0.3 * 1 +9 - 8 = 0.7
   R_1(s, Débordement de tampon, Blocage d'IP) = 0.7 * 0 - 0.3 * 1 +9 - 5 = 3.7
   R_1(s, Débordement de tampon, Isoler le véhicule) = 0.7 * 0 - 0.3 * 1 +9 - 10 = - 1.3
   R_1(s, Débordement de tampon, Arrêter le processus) = 0.7 * 0 - 0.3 * 1 +9 - 4 = 4.7
   R_1(s, Débordement de tampon, Aucune défense) = 0.7 * 1 - 0.3 * 0 +9 - 0 = 9.7

3.3.3 Forme normale du jeu

Une fois les stratégies de chaque joueur et les coûts associés identifiés, la forme normale du jeu est présentée dans le tableau 3.9.

Attaquant	**Administrateur**				
	Générer une alarme	**Blocage d'IP**	**Isoler le véhicule**	**Arrêter le processus**	**Aucune défense**
Inondation de requête	(-4.3, 4.3)	(-1.3, 1.3)	(-6.3, 6.3)	(-0.3, 0.3)	(4.7, -4.7)
Épuisement des ressources	(0.7, -0.7)	(3.7, -3.7)	(-1.3, 1.3)	(4.7, -4.7)	(9.7, -9.7)
Détournement de flux	(-2.3, 2.3)	(0.7, -0.7)	(-4.3, 4.3)	(1.7, -1.7)	(6.7, -6.7)
Manipulation du bus CAN	(1.7, -1.7)	(4.7, -4.7)	(-0.3, 0.3)	(5.7, -5.7)	(10.7, -10.7)
Exécution de code arbitraire	(-0.3, 0.3)	(2.7, -2.7)	(-2,3, 2.3)	(3.7, -3.7)	(8.7, -8.7)
Débordement de tampon	(0.7, -0.7)	(3.7, -3.7)	(-1.3, 1.3)	(4.7, -4.7)	(9.7, -9.7)

TABLE 3.9 – Forme normale du jeu

3.3.4 Analyse du graphe d'attaques

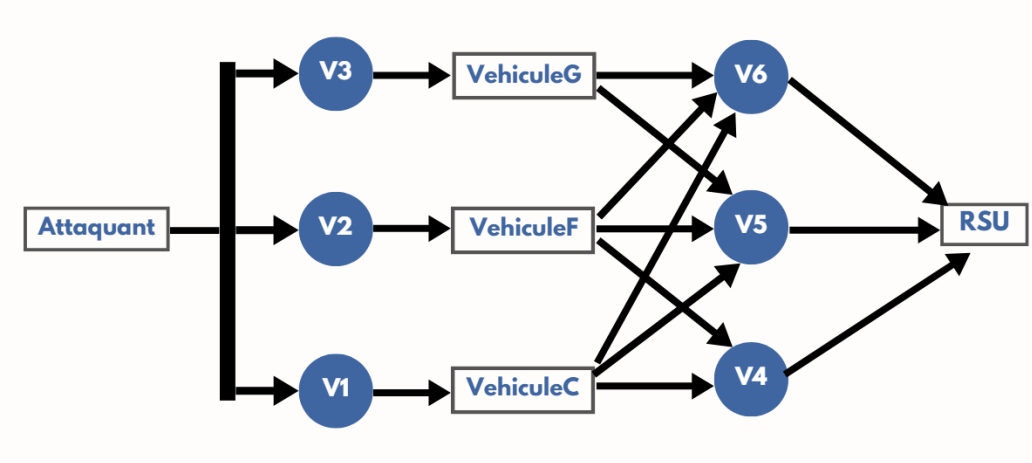

FIGURE 3.6 – Graphe d'attaques correspondant à la topologie du réseau étudiée.

La Figure 3.6 représente le graphe d'attaques que nous avons construit et qui va être analysé. Les rectangles représentent les dispositifs de la topologie représentée par la Figure 3.5 et les cercles représentent les vulnérabilités associer à chaque dispositif et que l'intrus utilise comme stratégies d'attaques.
Les chemins d'attaques sont :

Chemin 1 = {V1, VehiculeC, V4, RSU}
Chemin 2 = {V1, VehiculeC, V5, RSU}
Chemin 3 = {V1, VehiculeC, V6, RSU}
Chemin 4 = {V2, VehiculeF, V4, RSU}
Chemin 5 = {V2, VehiculeF, V5, RSU}
Chemin 6 = {V2, VehiculeF, V6, RSU}
Chemin 7 = {V3, VehiculeG, V5, RSU}
Chemin 8 = {V3, VehiculeG, V6, RSU}

3.3.5 Identification des clusters

Pour analysé le graphe de la Figure 3.6, nous avons utilisé l'algorithme K-Means pour former des clusters en fonction des données de coûts fournies pour chaque stratégie. Cet algorithme nous a permis de regrouper les différentes stratégies d'attaque en clusters distincts basés sur leurs coûts respectifs, offrant ainsi une vue claire de la répartition des coûts des stratégies d'attaque. Voici comment cela fonctionne :

— **Étape 1 : Données des Coûts des Stratégies**
Les données sont représentées sous forme de vecteurs où chaque vecteur contient les coûts associés à une stratégie donnée. Chaque coût est une paire de valeurs, l'une pour l'attaquant et l'autre pour l'administrateur, comme illustré dans la figure 3.7. Ces données peuvent être facilement manipulées et analysées en utilisant Python avec Jupyter Notebook. Ces outils permettent de représenter les vecteurs de coûts, de calculer les coûts totaux pour chaque partie, et de visualiser les résultats de manière interactive.

```
# Données des coûts des stratégies (sous forme de vecteurs)
data = np.array([
    [[-4.3, 4.3], [-1.3, 1.3], [-6.3, 6.3], [-0.3, 0.3], [4.7, -4.7]],   # Inondation de requête
    [[0.7, -0.7], [3.7, -3.7], [-1.3, 1.3], [4.7, -4.7], [9.7, -9.7]],   # Épuisement des ressources
    [[-2.3, 2.3], [0.7, -0.7], [-4.3, 4.3], [1.7, -1.7], [6.7, -6.7]],   # Détournement de flux
    [[1.7, -1.7], [4.7, -4.7], [-0.3, 0.3], [5.7, -5.7], [10.7, -10.7]], # Manipulation du bus CAN
    [[-0.3, 0.3], [2.7, -2.7], [-2.3, 2.3], [3.7, -3.7], [8.7, -8.7]],   # Exécution de code arbitraire
    [[0.7, -0.7], [3.7, -3.7], [-1.3, 1.3], [4.7, -4.7], [9.7, -9.7]]    # Débordement de tampon
])
```

FIGURE 3.7 – Données des coût de stratégies

— **Étape 2 : Normalisation des Données**
Le StandardScaler [45] est utilisé pour normaliser les données. Cette étape est cruciale pour le clustering, car elle met toutes les données sur une échelle comparable. La méthode fit_transform ajuste le scaler aux données et transforme les données en une distribution standard, comme illustré dans la figure 3.8.

```
# Normalisation des données
scaler = StandardScaler()
data_scaled = scaler.fit_transform(data)
```

FIGURE 3.8 – Normalisation des données

— **Étape 3 : Application de K-Means**
L'algorithme K-Means est initialisé avec 3 clusters en utilisant KMeans(n_clusters=3, random_state=0). Le paramètre random_state=0 assure la reproductibilité des résultats. La méthode fit est utilisée pour ajuster le modèle K-Means aux données normalisées. Les étiquettes de cluster attribuées à chaque point de données sont ensuite stockées dans labels, tel que représenté dans la figure 3.9.

```
# Application de K-Means
kmeans = KMeans(n_clusters=3, random_state=0)
kmeans.fit(data_scaled)
labels = kmeans.labels_
```

FIGURE 3.9 – Application Kmeans

— **Étape 4 : Resultats**
Les clusters formés :
Cluster 1 : Stratégies ['Inondation de requête']
Cluster 0 : Stratégies ['Épuisement des ressources', 'Manipulation du bus CAN', 'Exécution de code arbitraire', 'Débordement de tampon']
Cluster 2 : Stratégies ['Détournement de flux']

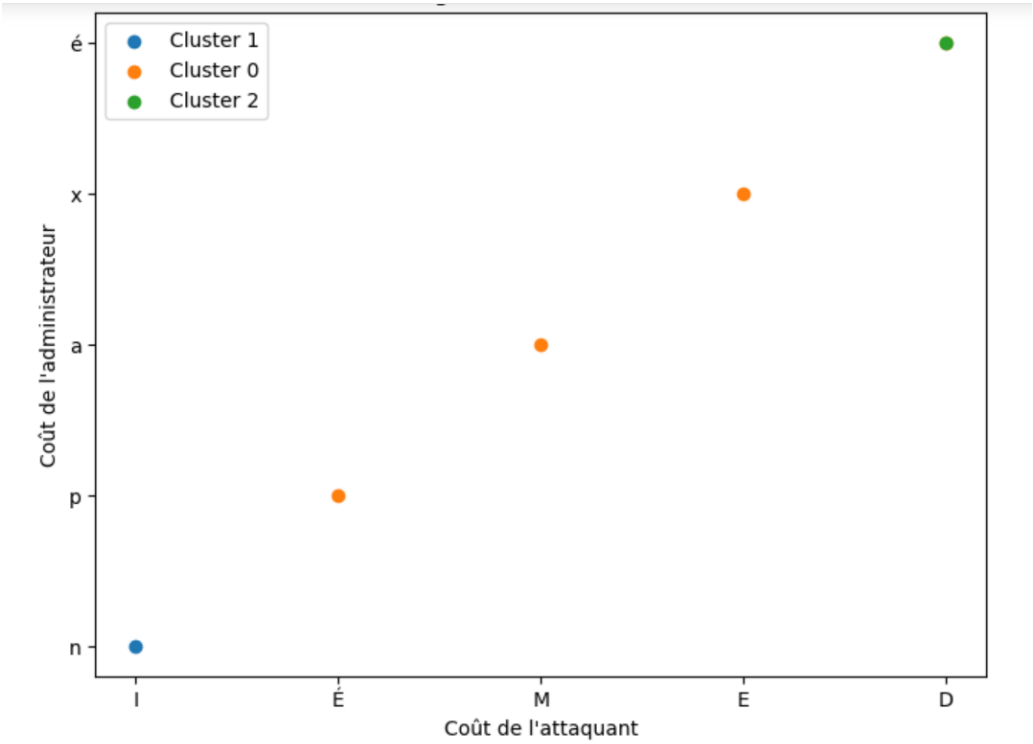

FIGURE 3.10 – Diagramme des clusters.

Le diagramme des clusters généré à partir de l'algorithme de K-Means dans la figure 3.10 présente une représentation visuelle des différents regroupements de stratégies d'attaques. Chaque point dans le diagramme représente une stratégie spécifique, avec sa position déterminée par ses coûts associés du point de vue de l'attaquant et de l'administrateur. Les clusters formés sont définis par des regroupements de stratégies similaires en termes de leurs coûts respectifs.

3.3.6 Analyse des clusters

Pour chaque cluster, nous allons calculer la moyenne des points de données appartenant à ce cluster par la formule (4) :

$$\mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} \mathbf{x}_i \tag{4}$$

où :

— N : nombre total de points de données.

— K : nombre de clusters.

— $\mathbf{x}_i$: le i-ème point de données.

— C_k : l'ensemble des indices des points de données appartenant au cluster k.

Étape 1 : Identification des points de données pour chaque cluster.
Nous avons les données suivantes pour chaque stratégie et les clusters auxquels elles appartiennent :

— Cluster 1 : Stratégies ['Inondation de requête'] - Points de données :

$$\begin{bmatrix} -4.3 & 4.3 \\ -1.3 & 1.3 \\ -6.3 & 6.3 \\ -0.3 & 0.3 \\ 4.7 & -4.7 \end{bmatrix}$$

— Cluster 0 : Stratégies ['Épuisement des ressources', 'Manipulation du bus CAN', 'Exécution de code arbitraire', 'Débordement de tampon'] - Points de données :

$$\begin{bmatrix} 0.7 & -0.7 \\ 3.7 & -3.7 \\ -1.3 & 1.3 \\ 4.7 & -4.7 \\ 9.7 & -9.7 \end{bmatrix}$$

$$\begin{bmatrix} 1.7 & -1.7 \\ 4.7 & -4.7 \\ -0.3 & 0.3 \\ 5.7 & -5.7 \\ 10.7 & -10.7 \end{bmatrix}$$

$$\begin{bmatrix} -0.3 & 0.3 \\ 2.7 & -2.7 \\ -2.3 & 2.3 \\ 3.7 & -3.7 \\ 8.7 & -8.7 \end{bmatrix}$$

$$\begin{bmatrix} 0.7 & -0.7 \\ 3.7 & -3.7 \\ -1.3 & 1.3 \\ 4.7 & -4.7 \\ 9.7 & -9.7 \end{bmatrix}$$

— Cluster 2 : Stratégies ['Détournement de flux'] - Points de données :

$$\begin{bmatrix} -2.3 & 2.3 \\ 0.7 & -0.7 \\ -4.3 & 4.3 \\ 1.7 & -1.7 \\ 6.7 & -6.7 \end{bmatrix}$$

Étape 2 : Calcul des moyennes pour chaque cluster

— Calcul pour le Cluster 1
Calcul de la moyenne pour chaque dimension :
Pour la première dimension (x) :

$$\text{Moyenne des x} = \frac{-4.3 + (-1.3) + (-6.3) + (-0.3) + 4.7}{5} = \frac{-7.5}{5} = -1.5$$

Pour la deuxième dimension (y) :

$$\text{Moyenne des y} = \frac{4.3 + 1.3 + 6.3 + 0.3 + (-4.7)}{5} = \frac{7.5}{5} = 1.5$$

Ainsi, le coût moyen pour le Cluster 1 est :

$$\begin{bmatrix} -1.5 & 1.5 \end{bmatrix}$$

— Calcul pour le Cluster 0
Calcul de la moyenne pour chaque dimension : Pour la première dimension (x) : Additionnons toutes les valeurs de la première colonne (x) :

$$\begin{aligned} &0.7 + 3.7 + (-1.3) + 4.7 + 9.7 + 1.7 + 4.7 + (-0.3) \\ &+ 5.7 + 10.7 + (-0.3) + 2.7 + (-2.3) + 3.7 + 8.7 + 0.7 \\ &+ 3.7 + (-1.3) + 4.7 + 9.7 \end{aligned}$$

Calculons ceci étape par étape :

$$0.7 + 3.7 = 4.4$$
$$4.4 + (-1.3) = 3.1$$
$$3.1 + 4.7 = 7.8$$
$$7.8 + 9.7 = 17.5$$
$$17.5 + 1.7 = 19.2$$
$$19.2 + 4.7 = 23.9$$
$$23.9 + (-0.3) = 23.6$$
$$23.6 + 5.7 = 29.3$$
$$29.3 + 10.7 = 40.0$$
$$40.0 + (-0.3) = 39.7$$
$$39.7 + 2.7 = 42.4$$
$$42.4 + (-2.3) = 40.1$$
$$40.1 + 3.7 = 43.8$$
$$43.8 + 8.7 = 52.5$$
$$52.5 + 0.7 = 53.2$$
$$53.2 + 3.7 = 56.9$$
$$56.9 + (-1.3) = 55.6$$
$$55.6 + 4.7 = 60.3$$
$$60.3 + 9.7 = 70.0$$

Maintenant, nous avons la somme totale pour la première dimension :

$$\text{Somme des x} = 70.0$$

La moyenne pour la première dimension (x) :

$$\text{Moyenne des x} = \frac{70.0}{20} = 3.5$$

Pour la deuxième dimension (y) :
Additionnons toutes les valeurs de la deuxième colonne (y) :

$$\begin{aligned} &-0.7 + (-3.7) + 1.3 + (-4.7) + (-9.7) + (-1.7) + (-4.7) + 0.3 \\ &+ (-5.7) + (-10.7) + 0.3 + (-2.7) + 2.3 + (-3.7) + (-8.7) + \\ &(-0.7) + (-3.7) + 1.3 + (-4.7) + (-9.7) \end{aligned}$$

Calculons ceci étape par étape :

$$\begin{aligned} -0.7 + (-3.7) &= -4.4 \\ -4.4 + 1.3 &= -3.1 \\ -3.1 + (-4.7) &= -7.8 \\ -7.8 + (-9.7) &= -17.5 \\ -17.5 + (-1.7) &= -19.2 \\ -19.2 + (-4.7) &= -23.9 \\ -23.9 + 0.3 &= -23.6 \\ -23.6 + (-5.7) &= -29.3 \\ -29.3 + (-10.7) &= -40.0 \\ -40.0 + 0.3 &= -39.7 \\ -39.7 + (-2.7) &= -42.4 \\ -42.4 + 2.3 &= -40.1 \\ -40.1 + (-3.7) &= -43.8 \\ -43.8 + (-8.7) &= -52.5 \\ -52.5 + (-0.7) &= -53.2 \\ -53.2 + (-3.7) &= -56.9 \\ -56.9 + 1.3 &= -55.6 \\ -55.6 + (-4.7) &= -60.3 \\ -60.3 + (-9.7) &= -70.0 \end{aligned}$$

Maintenant, nous avons la somme totale pour la deuxième dimension :

$$\text{Somme des y} = -70.0$$

La moyenne pour la deuxième dimension (y) :

$$\text{Moyenne des y} = \frac{-70.0}{20} = -3.5$$

Ainsi, le coût moyen pour le Cluster 0 est :

$$\begin{bmatrix} 3.5 & -3.5 \end{bmatrix}$$

— Calcul pour le Cluster 2
Calcul de la moyenne pour chaque dimension :
Pour la première dimension (x) :

$$\text{Moyenne des x} = \frac{-2.3 + 0.7 + (-4.3) + 1.7 + 6.7}{5} = \frac{2.5}{5} = 0.5$$

Pour la deuxième dimension (y) :

$$\text{Moyenne des y} = \frac{2.3 + (-0.7) + 4.3 + (-1.7) + (-6.7)}{5} = \frac{-2.5}{5} = -0.5$$

Ainsi, le coût moyen pour le Cluster 2 est :

$$\begin{bmatrix} 0.5 & -0.5 \end{bmatrix}$$

3.3.7 Identification des vulnérabilités prioritaires

Nous allons identifier le cluster avec le coût moyen le plus élevé pour l'administrateur qui est le cluster 0 et analyser les vulnérabilités des chemins associés.

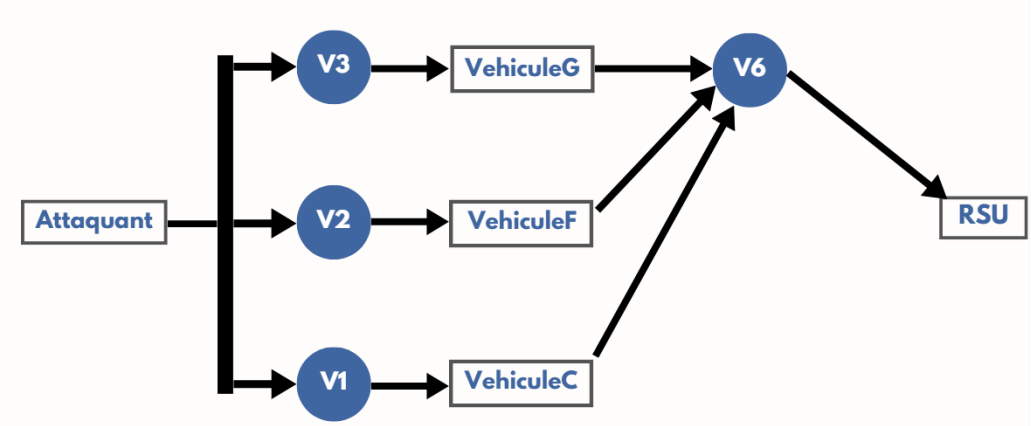

FIGURE 3.11 – Graphe d'attaques aprés la suppression de V4 et V5.

Nous allons calculer les coûts totaux des chemins d'attaque restant, qui peuvent être visualisés dans la figure 3.11. On va commencer par l'identification du risque de sécurité R par la formule(2), On a :

$l_3 = 2$, $l_6 = 2$, $l_8 = 2$, $m = 3$, $k = 4$, w= 0.5 donc :

$$R = \frac{1}{4} \times 0.5 + (1 - 0.5) \times \left(\frac{1}{2} + \frac{1}{2} + \frac{1}{2}\right)$$
$$R = 0.125 + 0.5 \times 1.5$$
$$R = 0.125 + 0.75$$
$$R = 0.875$$

Le tableau 3.10 présente le degré de chaque vulnérabilité où le nombre de vulnérabilité égale à 4 :
Pour calculer le coût de chaque chemin, nous allons utiliser la formule (3)

$$\text{Chemin3} = \frac{(1+3) \times 0.875}{4} = 0.875$$

Les différentes vulnérabilités	**Nombre d'arcs liés**
Éxécution de code arbitraire (V1)	1
Débordement de tempon (V2)	1
Manipulation de bus CAN (V3)	1
Épuisement des ressources (V6)	3

TABLE 3.10 – Tableau des degrés de chaque vulnérabilité.

$$\text{Chemin6} = \frac{(1+3) \times 0.875}{4} = 0.875$$

$$\text{Chemin8} = \frac{(1+3) \times 0.875}{4} = 0.875$$

3.3.8 Evaluation

Après avoir calculé les coûts des chemins d'attaque restants, nous constatons que les chemins 3, 6 et 8 ont tous le même coût, soit 0.875. Cela suggère que, du point de vue du coût, ces chemins sont équivalents.

En se référant au tableau 4.1, nous remarquons que la vulnérabilité V1 présente un niveau de risque élevé, tandis que les vulnérabilités V2, V3 et V6 ont un niveau de risque critique. Il est crucial de noter que la vulnérabilité V6 apparaît dans tous les chemins d'attaque restants, ce qui signifie qu'elle contribue de manière significative au risque global.

Par conséquent, il est recommandé de prioriser la correction (patching) de la vulnérabilité V6 plutôt que les vulnérabilités V1, V2 et V3. En concentrant nos efforts sur la correction de la vulnérabilité V6, nous pouvons réduire le risque associé à tous les chemins d'attaque restants, ce qui contribuera à renforcer la sécurité du système dans son ensemble.

Pour patcher la vulnérabilité V6 et atténuer l'épuisement des ressources, il est recommandé de déployer des outils de surveillance de la mémoire pour détecter rapidement toute fuite potentielle et surveiller de près les performances du système pour identifier les comportements anormaux. De plus, il est impératif de restreindre l'accès aux dispositifs vulnérables uniquement aux utilisateurs autorisés et de mettre en place des listes de contrôle d'accès pour filtrer le trafic suspect.

3.4 Discussion

Dans notre étude sur la sécurité des réseaux VANETs, nous avons adopté une approche intégrée basée sur l'analyse des vulnérabilités et la proposition de stratégies de sécurité. Notre objectif est d'équiper l'administrateur du réseau avec des outils pour atténuer les dommages causés par les attaques et renforcer la résilience du système dans son ensemble. Pour ce faire, nous avons développé une méthode s'appuyant sur l'utilisation d'un graphe d'attaques, qui cartographie les chemins qu'un attaquant pourrait emprunter pour compromettre le système. Notre approche consistait à analyser ce graphe afin de minimiser les pertes subies par l'administrateur et de lui fournir des indications pour renforcer la sécurité de son réseau. Cette analyse nous a permis de mettre en évidence les stratégies d'attaque les plus préoccupantes et de suggérer des contre-mesures appropriées.

Nous avons adopté deux méthodes pour analyser le graphe d'attaques. Premièrement, nous avons utilisé l'élimination des stratégies dominées dans le cadre de la théorie des jeux. Cette méthode implique de supprimer les stratégies d'attaque qui sont strictement dominées par d'autres, réduisant ainsi l'espace des stratégies à considérer. En parallèle, nous avons appliqué

l'algorithme K-means au graphe d'attaques. Cette approche nous a permis de regrouper les stratégies d'attaque similaires et d'identifier les clusters de stratégies les plus coûteux à contrer.

L'évaluation de notre approche a impliqué l'analyse des coûts des chemins d'attaque restants après avoir éliminé certaines vulnérabilités. En somme, notre travail offre une approche pour détecter et traiter les vulnérabilités dans les réseaux VANETs, combinant des techniques d'analyse des vulnérabilités, de théorie des jeux et d'apprentissage automatique.

3.5 Conclusion

Ce chapitre marque la derniére étape de notre projet, où nous avons présenté le fonctionnement de nos deux propositions en illustrant leurs applications sur deux exemples de graphe d'attaques. En résumé, notre travail a abouti à une approche novatrice pour renforcer la sécurité des réseaux VANETs.

CONCLUSION GÉNÉRALE ET PERSPECTIVES

Avec l'expansion des villes due à l'augmentation de la population, les individus se déplacent fréquemment sur de longues distances pour des raisons professionnelles et personnelles. Les Systèmes de Transport Intelligents sont devenus une nécessité majeure pour améliorer la gestion du trafic et réduire les temps de trajet. Les villes intelligentes de l'avenir résoudront efficacement les problèmes de circulation tels que les accidents, la notification d'urgence en temps réel et les embouteillages. La connectivité sans fil entre les véhicules offre une solution potentielle aux problèmes de transport majeurs.

Toutefois, les attaques de sécurité pourraient avoir un impact négatif sur les performances de diverses applications. Ainsi, la confidentialité et la sécurité des données partagées entre les différents composants des STI constituent une tâche technique importante. Les nœuds malveillants pourraient poser un risque de sécurité majeur car la plupart des applications STI impliquent la sécurité des personnes.

Dans ce contexte, nous avons proposé deux approches qui consistent à examiner des graphes d'attaques dans le but de minimiser les pertes subies par l'administrateur et de lui fournir des informations pour renforcer la sécurité de son réseau VANETs. Nous avons réorienté le défi de sécurité en un jeu stochastique fini et partiellement observable impliquant deux joueurs, sans coopération et à somme nulle. Cela nous a permis de tirer parti des techniques de résolution issues de la théorie des jeux et du machine learning, en éliminant les stratégies dominées pour la premiére approche et le clustering pour la deuxieme approche. Calculant par la suite les coût des chemins restants pour trouver la vulnérabilité a supprimé. Suite aux resultats de ces approches, nous concluons à son efficacité pour résoudre les problèmes d'analyse des graphes d'attaques et d'atténuer les vulnérabilités présentes dans les réseaux VANETs.

En conclusion, nous ouvrons des perspectives pour nos recherches futures. Parmi les pistes à explorer, il conviendrait d'étendre nos méthodes à des environnements VANETs plus complexes, intégrant des mobilités plus dynamiques et des scénarios de sécurité variés. De plus, une analyse approfondie de l'impact des différentes stratégies d'attaque et de défense sur les performances globales des réseaux VANETs pourrait orienter la conception de systèmes de sécurité plus robustes. Enfin, une comparaison avec d'autres méthodologies de sécurité existantes permettrait de mieux situer les avantages et les limites de notre approche par rapport à l'état de l'art.

BIBLIOGRAPHIE

[1] A.Perallos, U.Hernandez-Jayo, E.Onieva & I.J.G.Zuazola. "Intelligent Transport Systems : Technologies and Applications". In : *John Wiley Sons.* (2015), p. 24-25.

[2] P.J.Benghozi, S.Bureau & F.Massit-Folea. "L'Internet des objets. Quels enjeux pour les Européens ?" In : *hal-00405070* (2008). URL : `https://hal.science/hal-00405070/document`.

[3] J.Zhang, F.Y.Wang, K.Wang, W.H.Linand, X.Xu & C.Chen. "Data-Driven Intelligent Transportation Systems : A Survey". In : *IEEE Transactions on Intelligent Transportation Systems* 12.4 (2011), p. 1624-1639.

[4] S.Obana, N.Kadowaki & P.Davis. "Breakthroughs in Large-Scale Ad Hoc Wireless Networking and Application for Vehicle Safety". In : *7th International Conference on Mobile Data Management(MDM'06).IEEE* (2006), p. 88-88.

[5] D.Hahn, A.Munir & V.Behzadan. "Security and Privacy Issues in Intelligent Transportation Systems : Classification and Challenges". In : *IEEE Intelligent Transportation Systems Magazine* 13.1 (2019), p. 181-196.

[6] S.Quiguer. "Acceptabilité, acceptation et appropriation des Systèmes de Transport Intelligents : élaboration d'un canevas de co-conception multidimensionnelle orientée par l'activité." In : *Doctoral dissertation, Université Rennes 2* (2013).

[7] V.J.Hodge, S.O'Keefe, M.Weeks & A.Moulds. "Wireless Sensor Networks for Condition Monitoring in the Railway Industry : A Survey". In : *IEEE Transactions on Intelligent Transportation Systems* 16.3 (2014), p. 1088-1106.

[8] T.Mecheva & N.Kakanakov. "Cybersecurity in Intelligent Transportation Systems". In : *Computers* 9.4 (Decembre 2020), p. 83.

[9] W.Liang, Z.Li & H.Zhang. "Vehicular Ad Hoc Networks : Architectures, Research Issues, Methodologies, Challenges, and Trends." In : *International Journal of Distributed Sensor Network* 11.8 (2015).

[10] S.U.Rehman, M.A.Khan, A.Tanveer & L.Zheng. "Vehicular Ad-hoc Networks(VANETs) - An Overview and Challenges." In : *Journal of Wireless Networking and Communications* (2013).

[11] B.Mishra, P.Nayak & S.Behera. "Vehicular Adhoc Networks : A Survey." In : *ICCCS '11 Proceedings of the 2011 International Conference on Communication, Computing Security, ACM Digital Library* (2011), p. 590-595.

[12] A.Rosay. "Détection d'intrusions dans les objets connectés par des techniques d'apprentissage automatique : étude dans les domaines de l'éducation et des voitures connectées." In : *Le Mans Université* (2022), p. 80.

[13] K.P.Murphy. "Machine Learning : A Probabilistic Perspective". In : *MIT* (2012).

[14] T.Yuan, W.da Rocha Neto, C.E.Rothenberg, K.Obraczka, C.Barakat & T.Turletti. "Machine learning for next-generation intelligent transportation systems : A survey". In : *Transactions on Emerging Telecommunications Technologies* 33.4 (2022), e4427.

[15] S.Maldonado-Bascón, S.Lafuente-Arroyo, P.Gil-Jimenez, H.Gómez-Moreno & F.López-Ferreras. "Road-sign detection and recognition based on support vector machines." In : *IEEE Trans Intell Transp Syst.* 8.2 (2007), p. 264-278.

[16] G.L.Oliveira, W.Burgard & T.Brox. "Efficient deep models for monocular road segmentation". In : *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* (2016), p. 4885-4891.

[17] Z.Chen, N.Pears, M.Freeman & J.Austin. "Road vehicle classification using support vector machines". In : *IEEE International Conference on Intelligent Computing and Intelligent Systems.* 4. (2009), p. 214-218.

[18] A.Dominguez-Sanchez, M.Cazorla & S.Orts-Escolano. "Pedestrian movement direction recognition using convolutional neural networks". In : *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS.* 18.12 (2017), p. 3540-3548.

[19] X.Du, M.El-Khamy, J.Lee & L.Davis. "Fused DNN : a deep neural network fusion approach to fast and robust pedestrian detection." In : *IEEE winter conference on applications of computer vision (WACV).* (2017), p. 953-961.

[20] H.Xue, D.Q.Huynh & M.Reynolds. "SS-LSTM : a hierarchical LSTM model for pedestrian trajectory prediction." In : *IEEE Winter Conference on Applications of Computer Vision (WACV).* (2018), p. 1186-1194.

[21] W.Shao, Y.Zhang, B.Guo, K.Qin, J.Chan & F.D.Salim. "Parking availability prediction with long short term memory model". In : *Springer* (2018), p. 124-137.

[22] S.El-Tantawy, B.Abdulhai & H.Abdelgawad. "Design of reinforcement learning parameters for seamless application of adaptive traffic signal control". In : *Intelligent Transportation Systems* 18.3 (2014), p. 227-245.

[23] R.Zhang, A.Ishikawa, W.Wang, B.Striner & O.Tonguz. "Partially observable reinforcement learning for intelligent transportation systemsl". In : *arXiv preprint* (2018).

[24] Q.Huy, S.Mita & K.Yoneda. "A practical and optimal path planning for autonomous parking using fast marching algorithm and support vector machine". In : *IEICE TRANSACTIONS on Information and Systems.* 96.12 (2013), p. 2795-2804.

[25] H.Zhang, L.Dong, G.Gao, H.Hu, Y.Wen & K.Guan. "DeepQoE : a multimodal learning framework for video quality of experience (QoE)prediction". In : *IEEE Transactions on Multimedia* 22.12 (2020), p. 3210-3223.

[26] W.Wang, R.Lan, J.Gu, A.Huang, H.Shan & Z.Zhang. "Edge caching at base stations with device-to-device offloading". In : *IEEE Access* 5 (2017), p. 6399-6410.

[27] R.F.Atallah, C.M.Assi & M.J.Khabbaz. "Scheduling the operation of a connected vehicular network using deep reinforcement learning". In : *IEEE Transactions on Intelligent Transportation Systems* 20.5 (2018), p. 1669-1682.

[28] K.Koo, D.Moon, J.H.Huh, S.H.Jung & H.Lee. "Attack Graph Generation with Machine Learning for Network Security". In : *Electronics* 11.9 (2022), p. 1332. DOI : https://doi.org/10.3390/electronics11091332.

[29] N.Kadam & R.S.Krovi. "Machine Learning Approach of Hybrid KSVN Algorithm to Detect DDoS Attack in VANET". In : *International Journal of Advanced Computer Science and Applications.* 12.7 (2021).

[30] K.Rashid, Y.Saeed, A.Ali, F.Jamil & R.Alkanhel. "An Adaptive Real-Time Malicious Node Detection Framework Using Machine Learning in Vehicular Ad-Hoc Networks (VANETs)". In : *Sensors.* 23.5 (2023), p. 2594. DOI : https://doi.org/10.3390/s23052594.

[31] Y.Yu, L.Guo, Y.Liu, J.Zheng & Y.U.E.Zong. "An Efficient SDN-based DDoS Attack Detection and Rapid Response Platform in Vehicular Networks". In : *, IEEE Access* 6 (2018), p. 44570-44579. DOI : 10.1109/ACCESS.2018.2854567.

[32] P.Marc. "Théorie des Jeux Introduction". In : *Support de cours, Université de Lyon 1. https ://perso.liris.cnrs.fr/marc.plantevit/ENS/GameTheory/CM/Introduction.pdf* ().

[33] A.D.Madore. "Théorie des Jeux". In : *Support de cours, Université de Lyon 1* (2022).

[34] S.Tadelis. "GAME THEORY AN INTRODUCTION". In : *Princeton university press.* (2013).

[35] A.I.Sotvoldiyev & D.I.Ostonaqulov. "About Game Theory And Types Of Games". In : *Journal of Engineering and Technology* 23 (2023), p. 11-13.

[36] R.Bourlès & D.Henriet. "THÉORIE DES JEUX"". In : *Support de cours, University centrale Marseille.* (2016).

[37] M.Bowling & M.Veloso. "An Analysis of Stochastic Game Theory for Multiagent Reinforcement Learning". In : *Pennsylvania : School of Computer Science, Carnegie Mellon University* (2000), p. 0012.

[38] K.Bouafia & L.Hamza. "Game theory approach for analysing attack graphs". In : *Information and Computer Security.* 19.3-4 (2022), p. 305-320.

[39] L.Hamza, M.Yousfi & L.Bounehar. "Théorie des jeux et analyse des graphes d'attaques dans le contexte de l'IoT". In : *University of BEJAIA, mémoire de fin d'étude* (2023).

[40] M.Asadi. "Detecting IoT botnets based on the combination of cooperative game theory with deep and machine learning approaches". In : *Ambient Intelligence and Humanized Computing* 13.12 (2022), p. 5547-5561. DOI : https://doi.org/10.1007/s12652-021-03185-x.

[41] N.Phull, P.Singh, M.Shabaz & F.Sammy. "Enhancing Vehicular Ad Hoc Networks' Dynamic Behavior by Integrating Game Theory and Machine Learning Techniques for Reliable and Stable Routing". In : *Security and Communication Networks.* 2022.1 (2022), p. 4108231.

[42] URL : https://cve.mitre.org/cve/search_cve_list.html. Consulté le 21/04/2024.

[43] A.H.Anwar C.Kamhoua & N.Leslie. "A game-theoretic framework for dynamic cyber deception in internet of battlefield things". In : *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems : Computing, Networking and Services.* (2019), p. 522-526.

[44] F.Chen, D.Liu, Y.Zhang & J.Su. "A game-theoretic framework for dynamic cyber deception in internet of battlefield things". In : *Journal of Networks.* 5.5 (2010), p. 543.

[45] URL : https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html. Consulté le : 15/05/2024.

Résumé

L'intégration croissante de l'intelligence dans le système de transport devient de plus en plus critique. Bien que les attaques contre l'infrastructure de transport aient été rares jusqu'à présent, la prolifération des véhicules connectés accroît le risque de cyberattaques, notamment les attaques complexes telles que les attaques par déni de service distribué ciblées ou les tentatives d'infiltration sophistiquées visant à perturber les opérations. Ainsi, sécuriser les réseaux de véhicules ad hoc (VANETs) pour les véhicules individuels et les transports publics devient impératif. La sécurité de ces systèmes est essentielle pour assurer un transport efficace et sécurisé. Ce mémoire propose de nouvelles approches pour sécuriser les VANETs contre ces attaques complexes en analysant les graphes d'attaques à l'aide de la théorie des jeux et de l'apprentissage automatique. Ces méthodes facilitent la gestion de la sécurité pour les administrateurs en permettant l'identification et la mise en œuvre de stratégies défensives.

Mots clés : attaques complexes, VANETs, graphes d'attaques, théorie des jeux, apprentissage automatique, stratégies défensives.

Abstract

The increasing integration of intelligence into the transportation system is becoming increasingly critical. Although attacks on transportation infrastructure have been rare so far, the proliferation of connected vehicles increases the risk of cyberattacks, particularly complex attacks such as targeted distributed denial of service attacks or sophisticated infiltration attempts aimed at disrupting operations. Thus, securing vehicular ad hoc networks (VANETs) for individual vehicles and public transport becomes imperative. The security of these systems is essential to ensure efficient and safe transportation. This thesis proposes new approaches to secure VANETs against these complex attacks by analyzing attack graphs using game theory and machine learning. These methods facilitate security management for administrators by enabling the identification and implementation of defensive strategies.

Keywords : complex attacks, VANETs, attack graphs, game theory, machine learning, defensive strategies.