UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

MISBEHAVIOR AWARE ON-DEMAND INTRUSION DETECTION SYSTEM TO ENHANCE SECURITY IN VANETS WITH EFFICIENT ROGUE NODES DETECTION AND PREVENTION TECHNIQUES

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

DOCTOR OF PHILOSOPHY

By

ANIRUDH PARANJOTHI

Norman, Oklahoma

2023

Downloaded from SHAREOK. Persistent Handle URL: https://hdl.handle.net/11244/337169

MISBEHAVIOR AWARE ON-DEMAND INTRUSION DETECTION SYSTEM TO ENHANCE SECURITY IN VANETS WITH EFFICIENT ROGUE NODES DETECTION AND PREVENTION TECHNIQUES

A DISSERTATION APPROVED FOR THE
SCHOOL OF COMPUTER SCIENCE

BY THE COMMITTEE CONSISTING OF

Dr. Mohammed Atiquzzaman, Chair

Dr. Dean Hougen

Dr. Qi Cheng

Dr. Ronald Barnes

© Copyright by ANIRUDH PARANJOTHI 2023
All Rights Reserved.

To my late mother, Selvi

Acknowledgements

This work has been possible because of a number of individuals. First of all, I would like to express my gratitude to Dr. Mohammed Atiquzzaman for his patience, support, care, and continuous supervision. None of my achievements would have been possible without his guidance. Dr. Atiq has always been supportive and caring and has stood by me through the good and bad times.

I am truly grateful to my committee members: Dr. Cheng, Dr. Hougen, and Dr. Barnes, for their valuable advice and time.

I am thankful to my parents and friends. They have always been supportive and encouraged me with their best wishes.

Table of Contents

Acknowledgements **v**

List of Tables **viii**

List of Figures **x**

Abstract **xi**

1 Introduction **1**
1.1 Communication in VANETs 2
1.2 Applications of VANETs 4
1.2.1 ITA 4
1.2.2 CA 6
1.3 Background 6
1.3.1 Fog Computing in VANETs 7
1.3.2 Platooning in VANETs 9
1.4 Security in VANETs 14
1.5 Objective and Contributions of the Research 17
1.6 Organization of the Dissertation 18

2 Fog-based Rogue Nodes Detection in VANETs: False Information Attack **19**
2.1 Introduction 19
2.2 Related Works 23
2.3 System Model 26
2.3.1 Network Model 26
2.3.2 Traffic Flow Model 27
2.3.3 Attack Model 29
2.4 Proposed F–RouND Framework 29
2.4.1 Guard Node Selection 31
2.4.2 Cooperative Data Collection 33
2.4.3 Message Format 33
2.4.4 Speed and Density of Vehicles 34
2.4.5 Hypothesis Test to Validate the Vehicle Speed 36
2.4.6 Analysis of the Proposed F–RouND Framework 38
2.4.7 F–RouND Rogue Node Detection Algorithm 39
2.5 Performance Evaluation 40
2.5.1 Simulation Setup 40
2.5.2 Performance Metrics 42

2.6 Results . . . 44
2.6.1 Urban Scenario . . . 44
2.6.2 Highway Scenario . . . 48
2.7 Summary . . . 50

3 Fog-based Rogue Nodes Detection in VANETs: Sybil Attack 52
3.1 Introduction . . . 52
3.2 Related Work . . . 54
3.3 Proposed FSDV Framework . . . 57
3.3.1 Selection of Guard Node . . . 59
3.3.2 Density and Speed of Vehicles . . . 60
3.3.3 FSDV Algorithm . . . 61
3.4 Mathematical Model Analysis . . . 62
3.4.1 Analysis of Delay in FSDV . . . 62
3.4.2 Malicious Nodes Verification . . . 63
3.5 Performance Evaluation . . . 65
3.5.1 Evaluation of Speed Threshold in FSDV . . . 65
3.5.2 Simulation Setup . . . 66
3.5.3 Performance Metrics . . . 67
3.6 Results . . . 68
3.6.1 Urban Scenario . . . 68
3.6.2 Highway Scenario . . . 71
3.7 Summary . . . 74

4 Effect of Rogue nodes in Vehicular Platooning: Platoon Control Maneuver Attack 75
4.1 Introduction . . . 75
4.2 Related Works . . . 78
4.3 Proposed PMCD Framework . . . 79
4.3.1 Problem Description . . . 79
4.3.2 Rogue Node Detection Technique . . . 80
4.3.3 PMCD Algorithm . . . 81
4.4 Mathematical Model Analysis . . . 83
4.5 Performance Evaluation . . . 86
4.5.1 Simulation Setup . . . 86
4.5.2 Impact of the Rogue Node in the Platoon . . . 88
4.5.3 Performance Metrics . . . 89
4.6 Results . . . 90
4.7 Summary . . . 93

5 Conclusions and Future Directions 94

Bibliography 98

Author's List of Publications 114

List of Tables

2.1 Types of error and decisions in the null hypothesis testing 35
2.2 Parameters used in the simulation of the F–RouND framework . . . 41

3.1 Parameters used in Simulation of the FSDV Framework 66

4.1 Notations used in PMCD algorithm. 81
4.2 Parameters used in Simulation of the PMCD Framework 88

List of Figures

1.1 V2I Communication in VANETs. 1
1.2 V2I Communication in VANETs. 2
1.3 DSRC spectrum with one control channel and six service channels. . . 3
1.4 Two types of applications in VANETs: 1) ITA, 2) CA 4
1.5 Layered architecture of fog computing in VANETs. 7
1.6 An example scenario of platooning shows one-vehicle look-ahead communication in the platoon. All the vehicles are equipped with radar to measure the distance and speed of the preceding vehicle. 9
1.7 An example scenario of a platoon merge maneuver: a) platoon member M3 creates an intra-platoon gap instructed by the platoon leader (PL), so non-member vehicle V1 can join the platoon, and b) V1 joins the platoon and accomplishes the merging maneuver. 11
1.8 An example scenario of a platoon split maneuver: a) platoon leader (PL) sends split instructions to the platoon member M4, and b) M4 accomplishes the split maneuver by creating the gap, which splits one large platoon into two small platoons: platoon 1 and platoon 2. . . . 12
1.9 The Platoon member leave scenario: the intended vehicle requests the leader to exit the platoon. First, the leader splits the successors of the intended vehicle as a separate platoon (first split maneuver). Then, the second split maneuver takes place to make the intended vehicle exit the platoon. Finally, the two platoons are merged, and the gap is closed. 13

2.1 An example highway scenario of the traffic flow model. 28
2.2 Execution scenario of the F–RouND framework in the presence of a rogue node using the dynamic fog computing technique. 31
2.3 Hypothesis test of the F–RouND framework based on the average vehicle speed to determine the acceptance range values. 36
2.4 Simulated maps of the F–RouND framework from the City of Norman, USA: (a) urban and (b) highway scenarios. 40
2.5 Comparison of urban scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (a) data processing time and (b) PLR. . . 44
2.5 Comparison of urban scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (c) average throughput and (d) overhead. 45
2.5 Comparison of urban scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (e) TPR and (f) FPR. 46
2.6 Comparison of highway scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (a) data processing time and (b) PLR. 47

2.6 Comparison of highway scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (c) average throughput and (d) overhead. . . . 48
2.6 Comparison of highway scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (e) TPR and (f) FPR. 49

3.1 Execution scenario of FSDV in the presence of a rogue node using fog computing technique. . . . 57
3.2 Effect of dynamic speed threshold in the Sybil attack detection probability. . . . 65
3.3 Comparison of urban scenarios of the FSDV framework with PoW, IDS, and TM schemes: (a) data processing time and (b) PLR. 68
3.3 Comparison of urban scenarios of the FSDV framework with PoW, IDS, and TM schemes: (c) average throughput and (d) overhead. . . 69
3.3 Comparison of urban scenarios of the FSDV framework with PoW, IDS, and TM schemes: (e) TPR and (f) FPR. 70
3.4 Comparison highway scenarios of the FSDV framework with PoW, IDS, and TM schemes: (a) data processing time and (b) PLR. 71
3.4 Comparison highway scenarios of the FSDV framework with PoW, IDS, and TM schemes: (c) average throughput and (d) overhead. 72
3.4 Comparison highway scenarios of the FSDV framework with PoW, IDS, and TM schemes: (e) TPR and (f) FPR. 73

4.1 Interference caused by an unintended vehicle entered into a platoon in dense vehicle regions. . . . 79
4.2 Execution of platoon merge maneuver in the presence of the rogue node by splitting the platoon into sub platoons i.e. platoon A and platoon B. 80
4.3 An example scenario of the PMCD mathematical modeling to detect the rogue node in the platoon. . . . 84
4.4 A sample scenario of ten-vehicle platoon on a highway scenario: a) vehicles follow each other smoothly, and b) a rogue node enters the platoon during the platoon merge maneuver. . . . 86
4.5 Speed profile of vehicle stream with ten-vehicle platoon: a) vehicles speeds up and slow down, all vehicles follow each other properly and b) string stability is not maintained in the presence of a rogue node leads to distortion. . . . 87
4.6 Comparison of network throughput with Yang's approach and Michelle's approach by varying platoon size. . . . 89
4.7 Comparison of PLR with Yang's approach and Michelle's approach: (a) varying platoon size, (b) varying time [ms]. . . . 90
4.8 Comparison of collision ratio with Yang's approach and Michelle's approach by varying platoon size. . . . 91
4.9 Comparison of overhead with Yang's approach and Michelle's approach by varying platoon size. . . . 92

Abstract

Vehicular ad-hoc networks (VANETs) facilitate vehicles to broadcast beacon messages to ensure road safety. The goal behind sharing the information through beacon messages is to disseminate network state or emergency information. The exchange of information is susceptible to security attacks of different kinds. Amongst various problems to be solved in VANETs is the issue of rogue nodes and their impact on the network. Rogue nodes are malicious vehicles that are vicious to cause severe damage to the network by modifying or altering false data in beacon messages that could lead to catastrophic consequences like trapping a group of vehicles, road accidents, vehicle collisions, etc. This thesis discusses the problems associated with the security VANETs in the presence of rogue nodes.

We proposed three novel intrusion detection frameworks to detect the rogue nodes responsible for false information, Sybil, and platoon control maneuver attacks only by analyzing and comparing the beacon messages broadcast over the network. The novelty of our frameworks lies in containing network damage and securing VANETs from the harmful impact of rogue nodes. The proposed frameworks are simulated using SUMO, OMNET++, and VENTOS, and the results obtained have been presented, discussed, and compared to existing frameworks. Results show that the developed methods improve the systems' performance compared to existing methods even when the number of rogue nodes increases in the region.

Chapter 1

Introduction

Vehicular ad-hoc networks (VANETs) has evolved from mobile ad hoc networks (MANETs) with distinguished characteristics like high mobility and rapid change in topology. VANETs allow the vehicles to communicate with each other and exchange safety as well as non-safety information between the vehicles as messages [1,2]. Safety information includes road accident, roadblock, accident information, etc. Non-safety information includes tolling information, entertainment, etc. A report given by the association for safe and international road travel (ASIRT) concluded that nearly 1.25 million people die in road crashes each year, and distracted driving is one of the major reasons for road crashes [3]. As a result, VANETs emerged as the promising solution with a motivation to improve the road safety by reducing road accidents [3, 4].

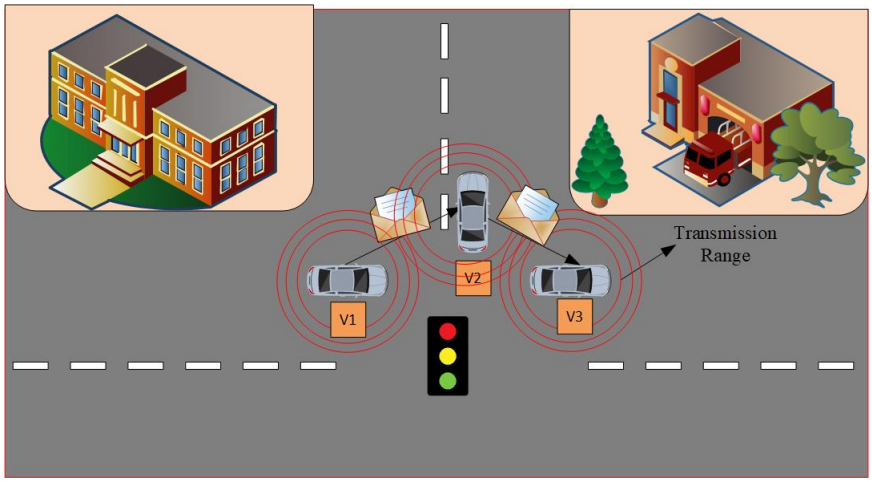


Figure 1.1: V2I Communication in VANETs.

1.1 Communication in VANETs

Vehicle to vehicle communication (V2V) and vehicle to infrastructure communication (V2I) are the two communication techniques used in VANETs. V2V allow the vehicles to communicate with each other directly using a multi-hop technique as long as the vehicles are in the transmission range of each other [3]. V2V communication is purely ad-hoc in nature since vehicles communicate with each other directly without infrastructure. Hence, it is less expensive when compared to V2I communication. Also, one main advantage of V2V communication is reduced communication overhead. However, it is not suitable for long-distance communication [5, 6]. A sample scenario of V2V and V2I communication is depicted in Fig. 1.1.

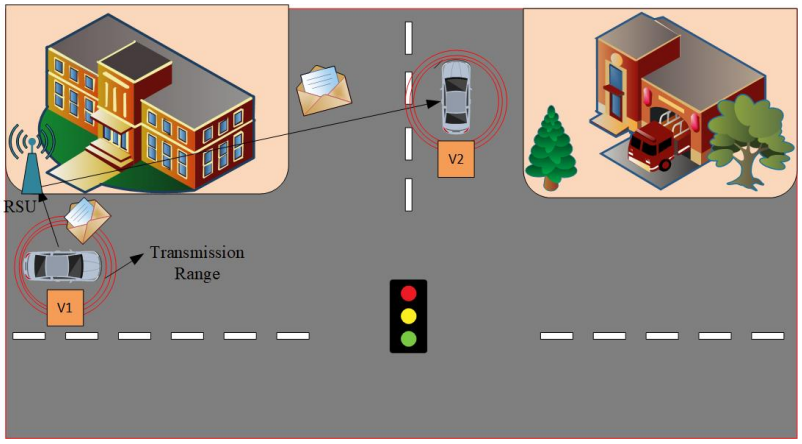


Figure 1.2: V2I Communication in VANETs.

V2I communication allows the vehicles to communicate with each other over a long distance using a multi-hop technique with the help of roadside infrastructure like road side units (RSU), etc. [3]. An advantage of V2I communication is providing support for the long distance communications. However, a considerable amount of communication overhead is involved in the transmission of messages. Enabled by a network of hardware, software, and firmware, the V2I technology is typically wireless and bi-directional. V2V and V2I communications are also known as short distance

and long distance communications respectively. A sample scenario V2I communication is depicted in Fig. 1.2.

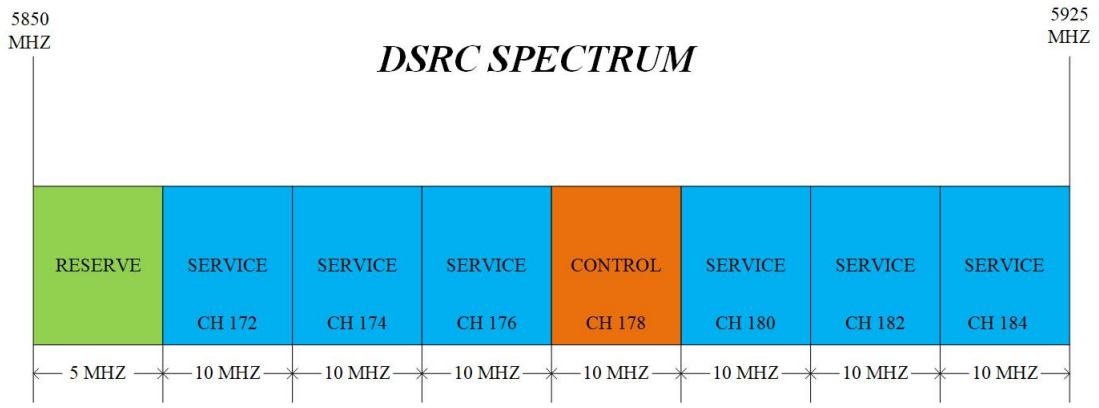


Figure 1.3: DSRC spectrum with one control channel and six service channels.

V2V and V2I communications in VANETs depends on the dedicated short range communication (DSRC) protocol. DSRC consists of a set of protocols for transmitting safety and non-safety information between vehicles and also between vehicles and RSU. Also, it employs the institute of IEEE 802.11p and IEEE 1609 standards for managing the performance of the network by wireless access in vehicular environment (WAVE) systems. The federal communication commission (FCC) set aside 75 MHz bandwidth of 5.9 GHz (5.850 GHz to 5.925 GHz) band for vehicular communication [7, 8], represented in Fig. 1.3.

DSRC has one control channel and six service channels for communication, in which the control channel is used to transmit safety information such as road accidents, natural hazards, etc. and the service channels are used to transmit non-safety information such as parking information, personal messages, etc. [3, 9] However, the performance of DSRC significantly decreases as the number of vehicles increases in the system. For example, regions like Manhattan are always congested with more number of vehicles at most all times resulting in the increase of load on DSRC spectrum, leading to instability in DSRC.

1.2 Applications of VANETs

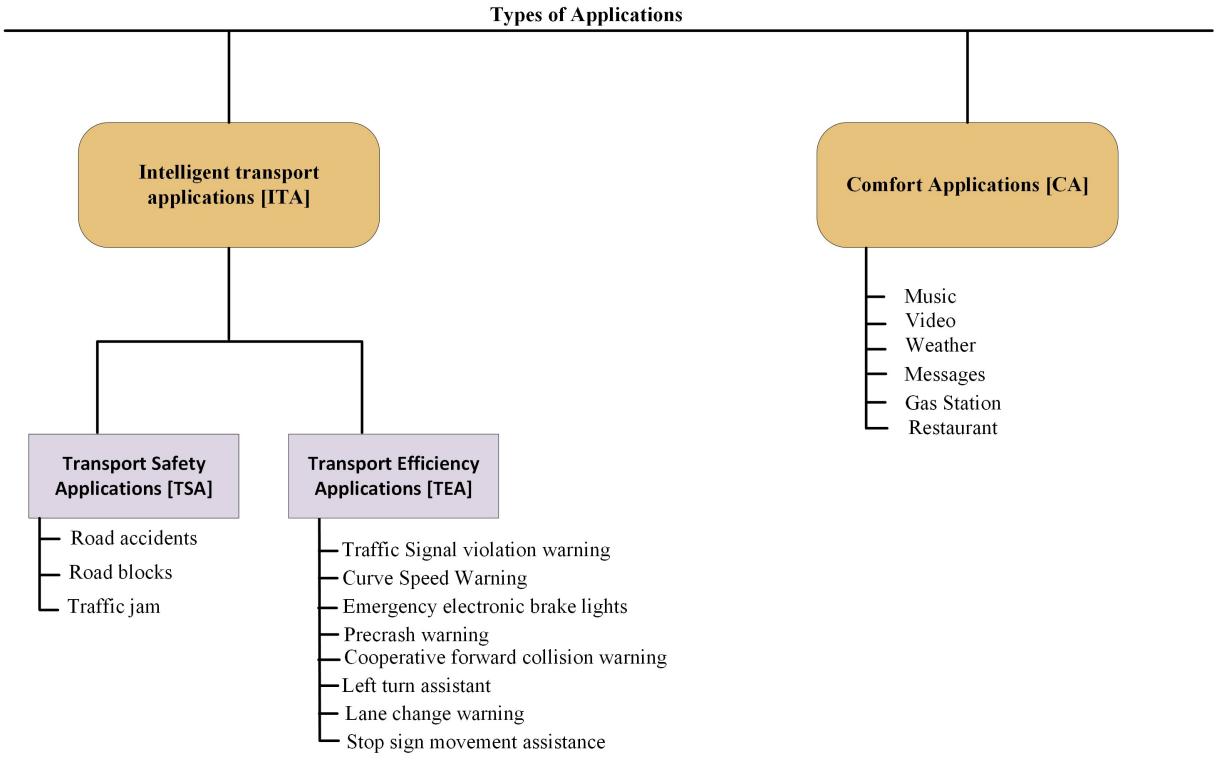


Figure 1.4: Two types of applications in VANETs: 1) ITA, 2) CA .

Development of cooperative and autonomus vehicles provides possible ways to develop applications for VANETS. These applications are categorized into two subgroups: 1) intelligent transport applications (ITA), and 2) comfort applications (CA) [3, 10], as shown in Fig. 1.4.

1.2.1 ITA

The aim of ITA is to ensure road safety by reducing accidents [3, 9]. Intelligent transport applications are classified into two types: 1) transport safety applications (TSA), 2) transport efficiency applications (TEA). Transport safety authority developed applications to avoid collisions, for example, if any accident occurs on the road, vehicular communication will be established, and messages are broadcast between

the vehicles in that region. Hence, the driver can select an alternate route. This application reduces congestion of vehicles in a specific region. Transport efficiency authority developed applications for prevention and vehicle safety communications (VSC) developed eight major applications to improve the road safety.

Traffic signal violation warning:

This application warns the driver who violated the traffic signal and the vehicles are required to make a complete stop of a vehicle during red light, flashing red light, stop sign, and railroad crossings.

Curve speed warning:

Curve speed warning alerts the driver about curve location, speed limit and level of curvature before the vehicle enters the area with the help of RSUs. Hence, drivers will be alarmed before potential dangers [3].

Emergency electronic brake lights:

This application is associated with collaborative adaptive cruise control (CACC). It provides automatic braking if the driver of the car does not react to the warning [3].

Precrash warning:

Precrash warning alerts the driver of nearby vehicles using a precrash signal if the collision of the car is unavoidable, so that neighboring drivers have more time to react. This avoids a fatal accident or pileup [3].

Cooperative forward collision warning:

This application helps the driver to avoid a rear-end collision by broadcasting warning signals to the drivers of impending collisions.

Left turn assistant:

Left turn assistant provides traffic information to the drivers and helps them to make the left turn at a signaled intersection.

Lane change warning:

Lane change warning warns the driver of the vehicle if an intended lane change will cause collision with a nearby vehicle.

Stop sign movement assistance:

This application warns the vehicle that it is about to cross through an intersection after being stopped by a stop sign on the road. It helps to prevent collision of vehicles in an intersection region.

1.2.2 CA

The role of CA is to make the journey more enjoyable for drivers and passengers [3] by providing infotainment applications. It enhances the road experience of the user. The applications include music, video, email access, weather information, gas stations, restaurants, and games. Also, it can access emergency assistance during mechanical emergencies by using appropriate applications [3].

1.3 Background

In this section, we provide some background concepts necessary for the reader to follow the ideas introduced later in this dissertation. Section 1.3.1 discusses fog computing in VANETs, and section 1.3.2 discusses platooning in VANETs.

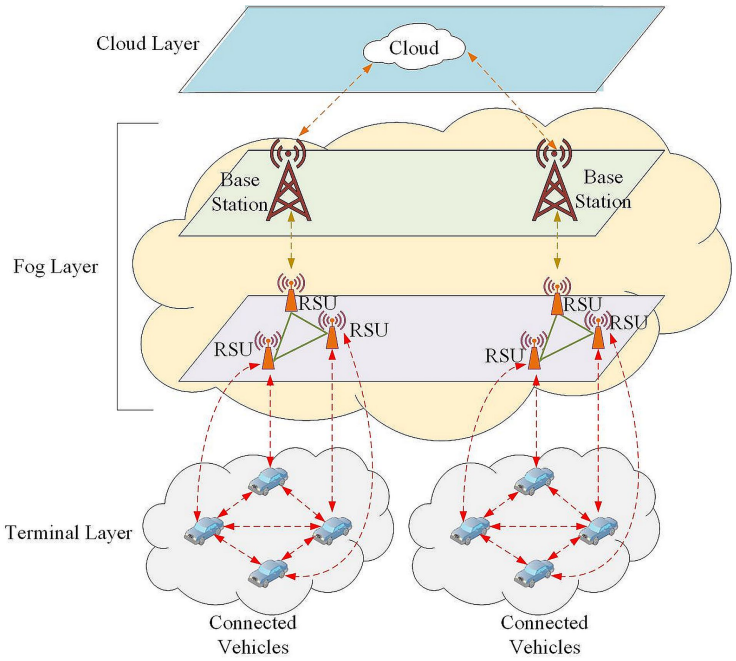


Figure 1.5: Layered architecture of fog computing in VANETs.

1.3.1 Fog Computing in VANETs

In a VANET environment, information is transmitted among the vehicles in terms of messages. However, the current V2V and V2I communications do not guarantee the message delivery due to the instability of DSRC, resulting in messages being dropped before reaching the destination. Thus, the emergence of a new paradigm (Fog computing) is essential to guarantee the message delivery. Fog computing (also known as edge computing) considered a new revolutionary way of thinking in wireless networking [11, 12]. It is an extension of cloud computing where computations are performed at the edge of the network. Any real-world objects which can acquire the properties storage, computing, and network connectivity can be formed as a fog node for a time period (t) resulting in rapid dissemination of messages between the vehicles [13]. In addition, fog computing also offers special services including location awareness, ultra-low frequency, and context information.

Being able to extend the cloud service to an edge of the network is the remarkable

characteristic of fog computing. By pooling the local services, fog computing enables control, computation, storage, and communication at the proximity of end users. By adding a resource rich layer between cloud and end devices, fog computing meets the challenges in high performance, interoperability, low latency, high reliability, mobility, and high security.

In fog computing, the extent of network transmission and the time required for data transfer are reduced as the network edge devices consume the data. The fog model can ease network bandwidth bottlenecks and adequately meet the needs of latency sensitive applications. It consists of many fog nodes, which includes: 1) virtualized edge data centers, 2) network edge devices, and 3) management systems. Fog nodes connect with users and end devices by wireless connections like Wi-Fi, 4G, Bluetooth, etc. in order to provide storage, computation, and computing services.

In vehicular fog environment, vehicles are considered as infrastructure to make up most utilization of computational resources and vehicular communications. The main objective of fog computing in VANETs is to utilize a large number of near-user edge devices or end-user clients to carry computation and communication. Besides the cloud characteristics, like providing application, storage and computing services to end users, fog computing differentiates itself from existing models with its dense geographical distribution and proximity to end users. Thus, fog computing provides low-latency at most all times in vehicular communications compared to existing techniques.

Fog-based layered architecture is shown in Fig. 1.5. Fog computing in VANETs consists of three layers: 1) Terminal layer, 2) Fog layer, and 3) Cloud layer.

Terminal Layer:

This layer closest to the physical environment and end user. It consists of various devices like smartphones, vehicles, sensors, etc. However, in VANETs only vehicles

are represented in the terminal layer. The vehicles are responsible for sensing the surrounding environment and transmitting the data to the fog layer for processing and storage.

Fog Layer:

Fog layer is located at the edge of a network. It consists of fog nodes, which includes access points, gateways, RSUs, base station, etc. Fog layer can be static at a fixed location or mobile on moving carriers such as in the vehicular environment [14, 15]. Also, they are responsible for processing the information received from the terminal device and temporarily store it or broadcast over the network.

Cloud Layer:

The main function of the cloud computing is to keep track of the resources allocated to each fog node and to manage interaction and interconnection among workloads on a fog layer.

1.3.2 Platooning in VANETs

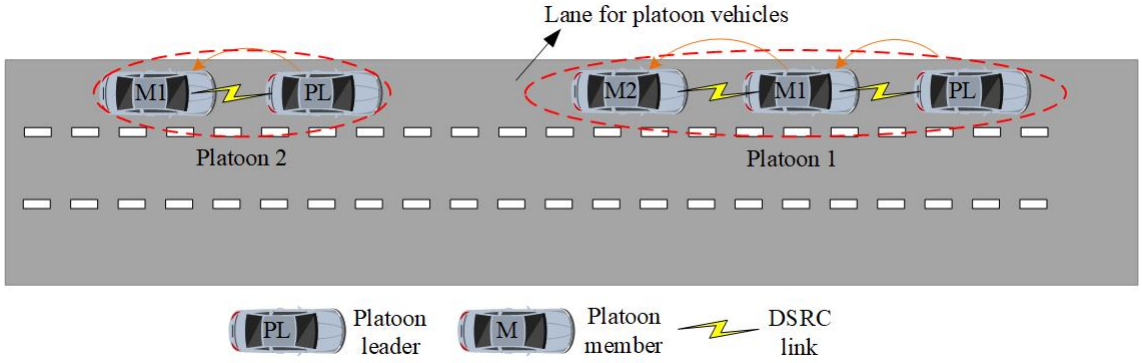


Figure 1.6: An example scenario of platooning shows one-vehicle look-ahead communication in the platoon. All the vehicles are equipped with radar to measure the distance and speed of the preceding vehicle.

Platooning is a road train consisting of a head vehicle called a leader, and several follower vehicles called members. In general, platoon-based driving is a cooperative driving pattern for a group of vehicles that shares common interests, where the vehicles follow the predecessor at a constant distance, speed, and acceleration [16,17,18]. The platoon leader determines the driving route and style and also ensures the platoon's stability at all times. All the vehicles are equipped with radar to measure the distance and speed of the preceding vehicle. A platoon has two basic maneuvers: 1) merge and 2) split.

In the merge, either a non-member vehicle requests the leader to join the platoon, and upon the leader's approval, the vehicle joins the platoon, or two platoons traveling on the same lane merge to form a single platoon. In the split, a platoon separates in a specific position to create two small platoons.

DSRC discussed in Section 1.1 supports inter-vehicle communications in the platoon. The platoon adopts centralized control technique where the platoon leader coordinates all communications. The members execute the instructions and send the requests from/to the leader [19]. In the platoon, the vehicles follow one-vehicle look-ahead communication, shown in Fig. 1.6. Each vehicle listens to beacon messages from its predecessor vehicle and then utilizes speed, distance, acceleration, etc., mentioned in the beacon message to maintain the stability of the platoon. The platoon leader listens to its preceding vehicle if present in the front of the platoon.

Basic Platoon Manuevers

This section explains how the basic maneuvers are executed in the platoon. Each platoon maneuver is performed by exchanging messages among the relevant vehicles. A special lane is reserved for platoon vehicles, and each platoon can perform different maneuvers to maintain the optimal size and stability of the platoon, usually dictated by RSU. Maneuvers can happen at any time; however, in most cases, more than one

maneuver is not permitted at a time, as it leads to catastrophic consequences.

1) Merge:

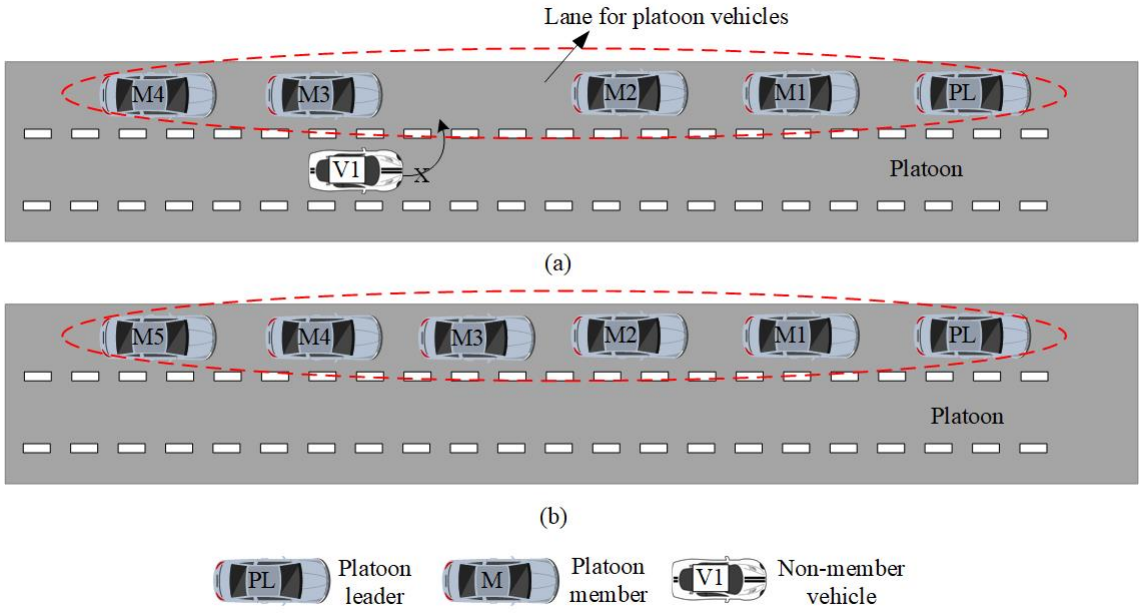


Figure 1.7: An example scenario of a platoon merge maneuver: a) platoon member M3 creates an intra-platoon gap instructed by the platoon leader (PL), so non-member vehicle V1 can join the platoon, and b) V1 joins the platoon and accomplishes the merging maneuver.

Merge maneuver allows non-member vehicles to join the platoon if the platoon size is less than the optimal size. Assume the platoon consists of five vehicles, where a non-member vehicle is interested in joining the platoon; first, the non-member vehicle can initiate a merge maneuver by sending the merge request to the platoon leader.

The platoon leader can either accept or deny the request. If the platoon size is more than the optimal size of the platoon or the leader is busy performing other maneuvers, it will reject the request. However, the latter case is called a weak reject. If the leader accepts the request, it sends the instruction to the relevant vehicle to create an intra-platoon gap through the unicast technique so the non-member can join the platoon. Once the gap is created, the relevant vehicle informs the non-member to perform a join operation to accomplish the merge maneuver. One such scenario is

shown in Fig 1.7, where PL, PM, and V1 are the platoon leader, platoon member, and non-member vehicles, respectively.

2) Split:

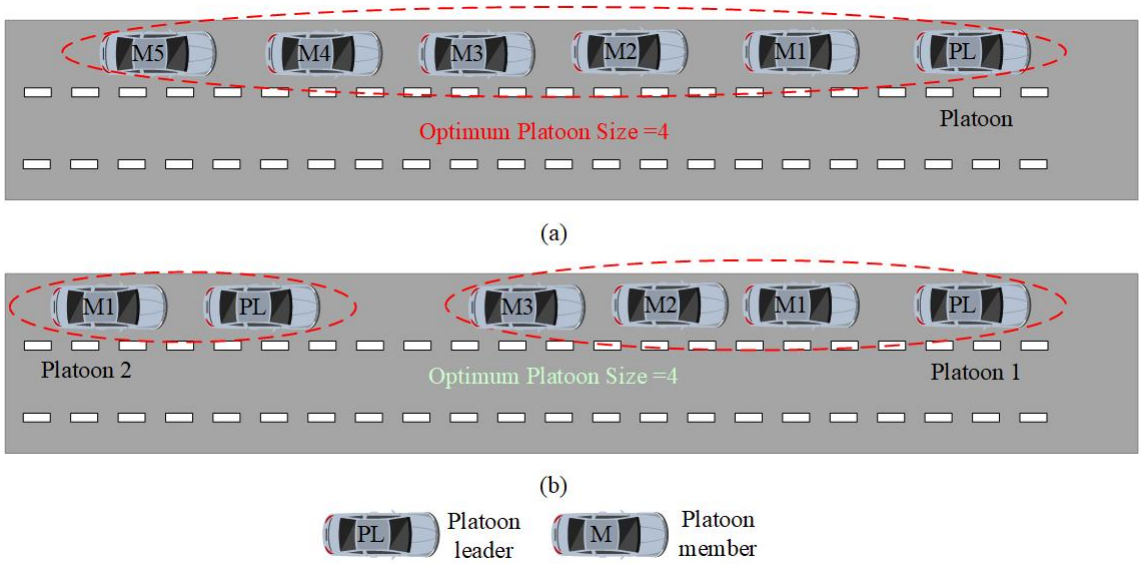


Figure 1.8: An example scenario of a platoon split maneuver: a) platoon leader (PL) sends split instructions to the platoon member M4, and b) M4 accomplishes the split maneuver by creating the gap, which splits one large platoon into two small platoons: platoon 1 and platoon 2.

Like merge, the split is also initiated by the platoon leader when platoon size exceeds the optimum size or if there is any instability in the platoon [18, 20, 21]. Split maneuver separates one platoon into two small platoons. Assume that our platoon size is six and the optimum platoon size is four; we need to split one platoon into two small platoons, platoon 1 and platoon 2, with sizes 4 and 2, respectively.

The platoon leader initiates the split by sending unicast split instructions to the relevant member vehicle and a multicast slowdown command to its successors. Upon accepting the request, the member vehicle creates the gap, which separates one large platoon into two small platoons, and informs the leader. Finally, the leader makes the member vehicle the new platoon leader and informs its successor vehicle to change the leader. One such scenario is shown in Fig. 1.8.

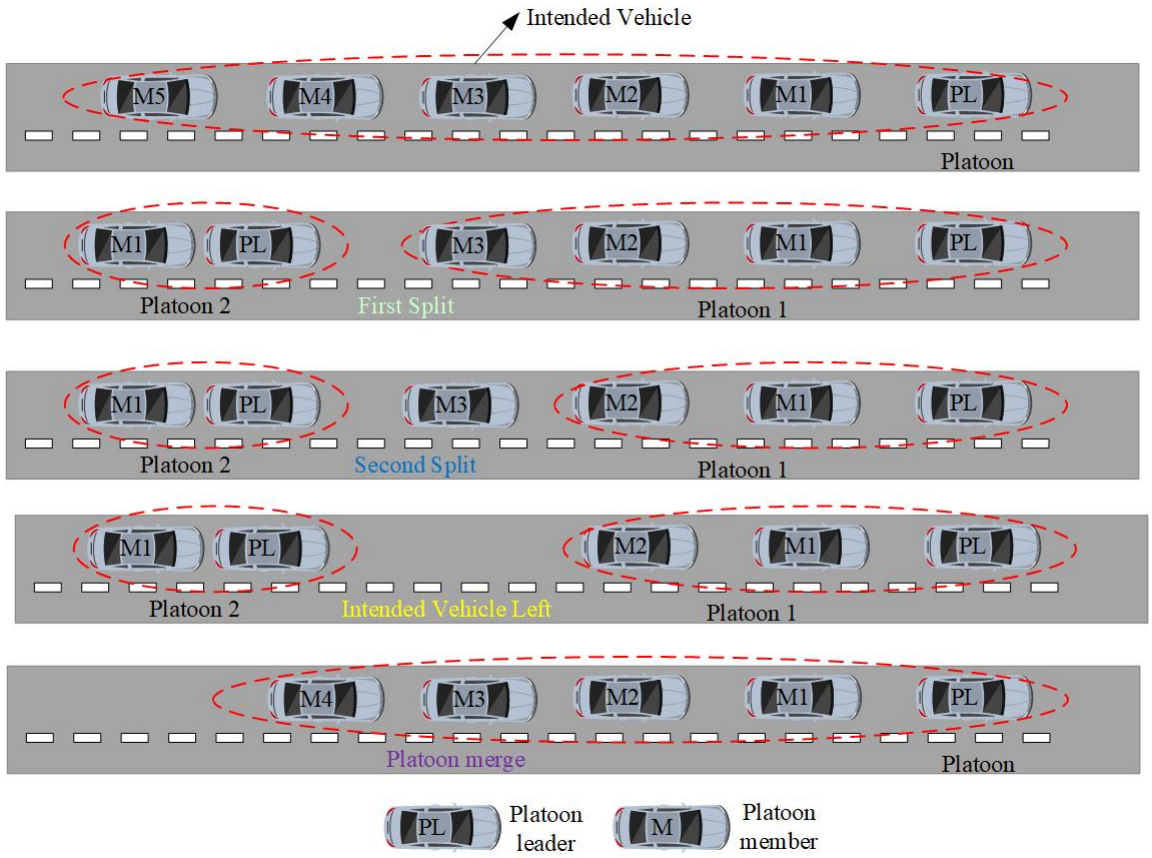


Figure 1.9: The Platoon member leave scenario: the intended vehicle requests the leader to exit the platoon. First, the leader splits the successors of the intended vehicle as a separate platoon (first split maneuver). Then, the second split maneuver takes place to make the intended vehicle exit the platoon. Finally, the two platoons are merged, and the gap is closed.

Scenarios in Basic Platoon Manuevers

Platoon leader and follower leave are the most common scenarios in basic platoon maneuvers. Both of these scenarios are used to exit the vehicle from the reserved platoon lane.

1) Platoon Leader Leave: The platoon leader leave scenario occurs when the leader needs to exit the platoon. Before the platoon leader leaves the platoon, it announces the second vehicle to be the leader, or the leader executes distributed leader selection algorithm, which allows followers to vote on a new platoon leader, and the one who gets the majority votes becomes the new leader.

2) Platoon Member Leave:

This scenario occurs when a platoon member needs to exit the platoon. The basic idea is to create enough space in the front unless it is the last vehicle in the platoon. Assume the platoon consists of six vehicles, and the third member needs to exit the platoon. First, the platoon member to exit the platoon (i.e., the intended vehicle) sends a request to the leader. If the platoon leader approves the request, it splits the successors of the intended vehicle as a separate platoon in the first split maneuver, and then the second split maneuver takes place to make the intended vehicle exit the platoon. Finally, the two platoons are merged, and the gap is closed, as shown in Fig. 1.9.

1.4 Security in VANETs

The biggest challenge in the deployment of VANETs is security. Security is the state of being free from threats or attacks. In VANETs, it is critical to define security well, as it is a highly mobile and dynamic network where topology changes frequently [22,23]. The vehicle communicates with each other through beacon message at regular intervals, consisting of basic information like speed, position, acceleration, deacceleration,

braking status, etc. The drivers make futuristic, life-critical decisions based on the information received in beacon messages from neighboring vehicles. Most security attacks target beacon messages. Therefore, security must guarantee that the attackers do not alter or insert malicious content in the exchanged beacon messages.

This thesis presents and implements an intrusion detection model for VANETs, which helps identify rogue nodes. Rogue nodes are malicious vehicles that are vicious to cause severe damage to the network by modifying or inserting false data in beacon messages or not participating in the communication. This could lead to catastrophic consequences like trapping a group of vehicles, road accidents, changing the intended route, etc. [24, 25, 26]

The rogue nodes can be identified by comparing and analyzing the values received from beacon messages. The vehicles which are close together will measure similar parameters in beacon messages and share them to develop a consensus about the road conditions [25, 27]. Each vehicle can compare the values being received from other vehicles and averages them to get an idea of the road conditions ahead. Moreover, each vehicle will compare the values received with the VANETs model and only accept them if they conform. In this way, inconsistent values in beacon messages can be detected, which in turn helps in identifying the rogue nodes.

Consider a rogue node that sends a false emergency braking message to all the neighboring vehicles in the region to create catastrophic consequences in the network. Now, this rogue node has to be a certain distance away from the targeted vehicles for them to fall for it; otherwise, the vehicles would experience or not experience the braking event themselves. When the rogue node sends the false emergency message, other vehicles nearby may have experienced the same emergency event and sent similar emergency messages. If such an emergency message comes from only one vehicle, it will be suspicious. Furthermore, suppose the rogue vehicle is sending such a false message further down the road. In that case, other vehicles in the vicinity should

have felt the effects of a real braking event, and their parameter values should have also decreased. If this is not the case, the emergence message is false.

This thesis discusses some of the most vulnerable security attacks in VANETs described below:

1. *False Information Attack:* Rogue nodes report an arbitrary event by modifying the values in the beacon messages with the vicious intent to cause damage to the network for their benefit. To create a greater impact, rogue nodes collectively modify speed values and the calculated value of density in their own beacon messages (Chapter 2). Under extreme conditions, the false information propagated in VANETs may cause the network to be paralyzed.

2. *Sybil Attack:* In a Sybil attack, rogue nodes create many fake identities, each with a different ID in the network, to create an illusion of traffic congestion ahead and thereby reroute the vehicles in the network to trap it or to cause catastrophic consequences, such as the collision of vehicles (Chapter 3). The IDs could have been either spoofed or stolen from compromised nodes. In general, Sybil attacks cause severe threats to bandwidth consumption and also harm network topology.

3. *Platoon Control Manuever Attack:* Each platoon performs different platoon control maneuvers, such as merge and splits, to maintain the optimal size and stability of the platoon. Platoon control maneuver attack, in Chapter 4, discusses strong interference caused by rogue nodes joining the platoon during the merging maneuver. Failing to acquire communication and accomplish platoon merge and split maneuvers leads to the distortion of the platoon by the platoon leader.

1.5 Objective and Contributions of the Research

The *objective* of this research is to monitor the network against various security attacks with the purpose of detecting rogue nodes responsible for broadcasting the bogus information in beacon messages or performing the malicious activity to cause catastrophic consequences, including vehicle collisions.

The key *contributions* of the dissertation are summarized as follows:

- Rogue node detection for VANETs is explained with different types of attacks and technical challenges.
- Fog-based rogue nodes detection is proposed to identify false information attack and Sybil attack in VANETs based on the traffic flow theory. Then, the performance of our proposed strategy is compared with some of the other strategies.
- We have developed an intrusion detection framework to identify the strong interference caused by a rogue node entering the platoon gap and disrupting platoon control maneuvers in highway scenarios
- We introduced the guard node in our framework, which is used to compare and analyze beacon messages from all vehicles in the region to detect rogue nodes in VANETs.
- We performed an extensive simulation by varying vehicular and network conditions to determine false information attacks, Sybil attacks, location spoofing attacks, and platoon control maneuver attacks caused by a rogue node during basic platoon maneuvers like merge and split using SUMO, OMNET++, and VENTOS simulators.

1.6 Organization of the Dissertation

The dissertation is organized as follows: In Chapter 2, a fog-based rogue node detection scheme to detect the false information attack is presented and followed by Sybil attack detection in VANETs in Chapter 3. Chapter 4 explains intrusion detection for platoon control maneuver attacks to establish cooperative driving in the presence of rogue nodes. Finally, Chapter 5 has concluding remarks and future research directions.

Chapter 2

Fog-based Rogue Nodes Detection in VANETs: False Information Attack

2.1 Introduction

Recent advancements in wireless technology have brought a significant development in VANETs, which are considered as a state–of–the–art technology in ITSs in terms of enhancing road safety by reducing the number of accidents and optimizing the traffic flow. DSRC provides wireless communication capability that permits vehicles to communicate with each other using V2V and RSUs via V2I techniques [3, 22]. The vehicles are equipped with OBUs for transmitting and receiving messages, including beacon messages. VANETs facilitate vehicles to broadcast beacon messages to disseminate the network state or emergency information to reduce road accidents and traffic congestion [28, 29, 30]. However, malicious vehicles acting as legitimate vehicles, also known as rogue nodes, may broadcast malicious information, such as false congestion and collision warning for their own benefits [31, 32]. Rogue node detection plays a crucial role in establishing a secure VANET environment because misleading/false information in beacon messages results in changing the normal behavior of vehicles, which may lead to catastrophic consequences, including vehicle collision [33, 34, 35]. Therefore, efficient rogue node detection is crucial in containing network damage.

Previous authors used either cryptography, trust scores, or past vehicle data to detect rogue nodes. Al-Otaibi et al. [36] proposed a cryptography–based Intrusion Detection Scheme (IDS) using fog computing to detect rogue nodes. This scheme

was known as Fog–IDS. RSUs act as fog nodes transmit public keys to the vehicles in its communication range. The vehicles use private–public key pairs for encrypting messages before being transmitted to the RSUs. Once the messages are received from the vehicles, the RSUs authenticate the key pairs and broadcast the messages to all other vehicles in the region. However, this approach [36] has high processing delay and overhead in detecting rogue nodes when the RSUs are overloaded or not available in the region. Zaidi et al. [37] presented an IDS to detect rogue nodes based on past vehicle data. Each vehicle collects historical information about all other vehicles in the region. Once sufficient details have been collected, the vehicles utilize their OBUs to combine the data and find rogue nodes in the region. The proposed scheme [37] has limitations, such as high delay and overhead. Ahmad et al. [38] proposed a trust–based scheme called Trust Evaluation and Management (TEAM). The TEAM framework comprises three different trust models for detecting rogue nodes: entity, data, and hybrid–oriented trust models. The entity–oriented model performs better compared to the data and hybrid–oriented trust models in detecting false data exchanged between vehicles due to the presence of high trustworthy vehicles. However, the framework [38] encounters high delay and FPR in detecting rogue nodes when the number of vehicles increases in the region.

To address the limitations of the existing rogue node detection schemes, we propose herein a Greenshields traffic flow based–statistical framework for VANETs, called Fog–based Rogue Node Detection (F–RouND). The F–RouND framework is based on fog computing which dynamically utilizes the OBUs of all vehicles in the region for rogue node detection. The proposed framework employs a twofold process in rogue node detection. First, we use the guard node concept to detect rogue nodes. The guard node is the vehicle with more neighboring vehicles in its communication range than all other vehicles and creates a dynamic fog computing layer by utilizing the OBUs of all the vehicles in the region. The fog computing layer is then used to

compare the speed of all the vehicles to the detect rogue nodes in the region. Second, the guard node performs a hypothesis test to validate whether the rogue nodes are correctly identified. Upon successful validation, the guard node broadcasts the rogue node information to all the vehicles in the region. The F–RouND framework exploits fog computing, which is characterized by low latency and high bandwidth, and performs computations at the network edge [13, 39].

The *novelty* of the work proposed herein providing low processing delays and FPR at high vehicle densities. In addition, our F–RouND framework does not depend on any roadside infrastructures, such as RSUs, or trust scores or past vehicle data in rogue node detection. The *difference* between the F–RouND framework and the existing schemes [36, 37, 38] is that each vehicle uses its OBU or RSU to detect rogue nodes. RSUs are not uniquely deployed in all VANET regions. The absence of RSUs yields high processing delay and FPR. The OBUs of individual vehicles are resource–constrained and cannot be used to process a large number of beacon messages received from all the vehicles in a region because it results in a high delay. VANETs are highly dynamic in nature. The significant delay associated with the rogue node detection may lead to severe network damage. In the F–RouND framework, the OBUs of all the vehicles are combined while creating the dynamic fog layer, which increases the computation power compared to individual OBUs, resulting in low processing delays and FPR.

Our *objective* is to reduce the latency in detecting rogue nodes, increase the True Positive Rate (TPR), and decrease the FPR at high vehicle densities. We considered three existing rogue node detection schemes for comparison: Fog–IDS [36], IDS [37], and TEAM [38]. A tradeoff always exists when choosing the appropriate rogue node detection schemes for comparison. Choosing from among the existing rogue node detection schemes in VANETs is a challenging task that should consider factors such as the approaches used in rogue node detection, network size, type of data that will be

transmitted, and mobility models. Based on the aforementioned factors, the schemes presented in [36, 37, 38] are the best–known schemes for rogue node detection and have been used by most researchers in the recent times for comparison.

The performance of the F–RouND framework was evaluated via simulations using OMNET++ and SUMO simulators for highway and urban scenarios with up to 4000 vehicles and 40% rogue nodes. Results showed that the F–RouND framework ensures 45% lower processing delays, 12% lower overhead, and 36% lower FPR at the urban scenario and 44% lower processing delays, 10% lower overhead, and 32% lower FPR at the highway scenario compared to the [36,37,38] schemes. Overall, our framework performs up to 38% better than the existing rogue node detection schemes even when the number of rogue nodes increases by up to 40% in the region.

The *contributions* of this chapter are as follows:

1. We proposed a framework that uses statistical techniques and Greenshields traffic model to detect rogue nodes at all vehicle densities in the urban and highway scenarios.

2. We introduced the guard node in our framework, which uses an OBU–based fog computing technique to compare and analyze beacon messages from all vehicles in the region and to perform an extensive hypothesis test to accept or reject data.

3. The proposed framework utilizes only vehicle speed values in beacon messages and does not depend on either trust score, cryptography, or past vehicle data in rogue node detection.

4. We performed an extensive simulation by creating a dynamic fog layer under varying vehicular and network conditions to determine false information broadcasted in both urban and highway scenarios.

2.2 Related Works

Security is an important issue in the VANET environment. Thus, detecting rogue nodes broadcasting false information through beacon messages plays a crucial role in establishing a secure environment [40, 41]. In VANETs, messages are broadcasted to neighboring vehicles using either broadcasting or multihop techniques. The users make life–saving decisions based on the information received from other vehicles [42]. Therefore, the messages received from the rogue nodes in dynamic vehicular networks may cause havoc by broadcasting false congestion or accident information to the neighboring vehicles. This section presents an overview of three main techniques used in existing approaches to detect rogue nodes in the region: cryptography, trust–based schemes, and past vehicle data.

Arshad et al. [43] proposed a beacon–based trust scheme to detect false messages in VANETs. Initially, the trust values of all vehicles are assigned to be 0; then, based on the data correctness, positive or negative trust values are assigned. Positive and negative trust values represent the normal and abnormal behaviors of the neighboring vehicles, respectively. When the calculated trust of any vehicle reaches a predefined threshold limit known as rogue nodes, the information is broadcasted to all the vehicles in the region. However, [43] suffers from high Packet Loss Ratio (PLR) and FPR. Liang et al. [44] presented a feature extraction algorithm for detecting false messages broadcasted by rogue nodes; this algorithm adopts past vehicle data for training and testing the data received from other vehicles in the network. The feature extraction algorithm comprises two modules: a feature extraction module and a classifier module. The feature extraction module extracts the vehicle flow and position from the received vehicle data and sends the information to the classifier module for verification. Upon receiving the values of the vehicle flow and position, the classifier module compares the received values against the past vehicle data to detect the rogue nodes.

This approach [44] has a high delay in data extraction and processing at all vehicle densities.

Sedelmaci et al. and Ahmed et al. [45,46] proposed trust–based schemes to detect rogue nodes; RSUs were used to compute trust scores and detect rogue nodes based on the calculated trust score. Zhang et al. and Shams et al. [47,48] illustrated the rogue node detection mechanism based on the Support Vector Machine (SVM) and Dempster–Shafer Theory (DST) to resist false messages. The frameworks [47, 48] comprise a local trust module and a vehicle trust module. The local trust module uses an SVM–based classifier to detect false messages, while the vehicle trust module uses DST to derive the comprehensive trust value for all vehicles. The results of both the local and trust modules are then combined to find the rogue nodes in the region. However, the approaches [45, 46, 47, 48] have low TPR and high overhead at high vehicle densities.

Mundhe et al. [49] illustrated a cryptographic–based message authentication scheme to detect false messages propagated among vehicles in VANETs. The proposed framework [49] employs a ring signature to generate the public and private keys for each vehicle in the network. When the sender broadcasts a beacon message to the neighboring vehicles, a ring signature gets generated and transmitted along with the message. The receiver accepts this message upon successful verification, which helps identify whether the message is modified in the middle. However, a significant delay is associated with the generation of the private and public keys for each vehicle when the RSUs are overloaded or not available in the region. Yang et al. [50] proposed a machine learning algorithm that verifies whether the data received from the vehicles in the region are valid by comparing them with the past vehicle data using classification and decision tree techniques. This approach [50] suffers from low TPR and high FPR when the number of vehicles increases in the region.

Tripathi et al. [51] and Nandy et al. [52] proposed trust–based intrusion detection

frameworks for rogue node detection. The models proposed in [51, 52] assigns a trust value to all the vehicles in the region based on their behavior. The vehicles that either drop or alter the message are considered rogue nodes with negative trust values. Vehicles other than the rogue nodes are trustworthy vehicles with positive trust values. The trust values of the vehicles are maintained in the score table, which gets updated whenever messages are broadcasted by the vehicles. Each vehicle has a copy of the score table containing information about all the vehicles in the region. Thus, the messages received from the rogue nodes are ignored to contain network damage. As such, the frameworks proposed in [51, 52] suffer from high PLR and FPR in detecting the rogue nodes.

Liu et al. [53] proposed a Bayesian interference–based traffic model to identify false messages in VANETs. Based on the Bayesian approach, the model [53] calculates the likelihood of traffic patterns for the future, which is further used to verify whether events, such as road accidents, reported by the vehicle have happened. If the model identifies the reported event as wrong, the corresponding vehicles that broadcast the messages are considered rogue nodes. The information regarding rogue nodes is broadcasted to all vehicles in the region. Manimaran et al. [54] presented a framework for Named Data Networking (NDN)–based VANETs. The framework analyzes the messages received from all other vehicles with past vehicle data and predefined rules to detect rogue nodes. Once the false messages are detected, they are separated from the genuine messages under various test cases, resulting in trustworthy messages being broadcasted to all other vehicles in the region. In [53, 54], the OBUs of each vehicle were used to verify whether the messages were valid. The OBUs are highly resource–constrained; hence, this approach is not suitable for highly dense regions such as Manhattan and other downtown environments.

Zhou et al. [55] proposed a distributed collaborative intrusion detection framework that stores and compares past vehicle data to identify the false messages broadcasted

by the rogue nodes in VANETs. The scheme proposed in [55] employs a clustering technique to segregate the vehicles based on the reputation state and behavior; then, the normal driving characteristics are compared with the individual clusters to detect malicious behaviors. The vehicles associated with the clusters exposing malicious behaviors termed rogue nodes are then ignored to contain the network damage. This approach [55] suffers from high processing delays in creating clusters and high PLR at high vehicle densities.

To overcome the limitations of the existing rogue node detection schemes [36-38, 43-55], we propose herein the F–RouND framework, which does not depend on either trust scores, cryptography, or past vehicle data but, instead adopts the fog computing technique, Greenshields traffic flow theory, and the statistical model to detect rogue nodes in the region. The hypothesis test used for validating the result of the guard node, which declares whether the vehicle is rogue, increases the efficiency and performance of our framework compared to existing schemes even when the percentage of rogue nodes in the region is 40%. In addition, the F–RouND framework does not depend on any roadside infrastructures, including RSUs, in the rogue node detection.

2.3 System Model

In this section, we illustrate the mechanism and models, such as the network model, traffic flow model, and attack model, adopted in this study.

2.3.1 Network Model

In VANETs, vehicles communicate with each other through messages. Vehicles are equipped with various devices, such as GPS, Radar, and OBU, to disseminate speed, position, acceleration, braking status, etc. to the neighboring vehicles. V2V and V2I

communication is used to transmit messages using either multihop or broadcasting techniques. Based on the received messages from neighbors, every vehicle adjusts their speed, acceleration, etc. to maintain the network state and behavior. We classify the vehicles in the F–RouND framework into two categories: honest, and rogue nodes. Honest nodes are the vehicles broadcasting genuine messages to the neighboring vehicles, while rogue nodes are those injecting false data before broadcasting it to the network.

We also employ the guard node concept to detect rogue nodes. The guard node is the most trustworthy vehicle located in the center of the network and creates a dynamic fog layer to compare and analyze the vehicle speed in the beacon messages to identify false messages broadcasted by the rogue nodes in the region. The OBUs of individual vehicles are resource–constrained; thus, guard nodes utilize the OBUs of all the vehicles for creating the dynamic fog layer. Our F–RouND framework works efficiently even if there are multiple guard nodes in the region. However, the adoption of the fog computing technique provides a high computation power to the guard node, resulting in a lower delay and a high scalability at all vehicle densities, including highly dense regions, such as downtown environments. Thus, there is no need for multiple guard nodes to detect rogue nodes in any given circumstances. The working principle of the guard node, including the guard node selection, is presented in Section 2.4.

2.3.2 Traffic Flow Model

We adopted the concept of Greenshields traffic flow model to model the traffic flows in urban areas and highways. Greenshield model is a fairly accurate and simple model for predicting the traffic flows observed in real–world scenarios. It works under the assumption of density (ρ) and speed of vehicles (S) negatively correlated [56]. The relationship between speed and density is defined as follows:

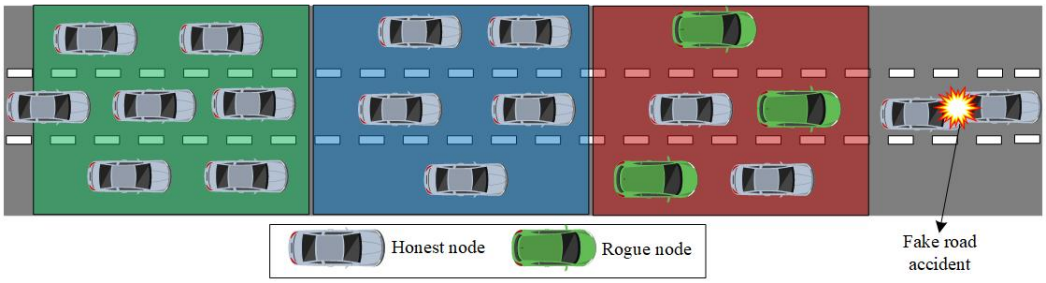


Figure 2.1: An example highway scenario of the traffic flow model.

$$S \propto \frac{1}{\rho} \tag{2.1}$$

$$S = C.\frac{1}{\rho} \tag{2.2}$$

where, C is a constant that depends on the communication range of the vehicle. As the speed and density of the vehicles are negatively correlated, the density increases when the vehicle speed decreases in the region and vice versa. The maximum density is the point at which the speed becomes zero, known as (ρ_{max}), and the maximum speed is when the density becomes zero, known as (S_{max}).

For example, consider the highway scenario depicted in Fig. 2.1, where vehicles are traveling at high speed (i.e., 60–70 mph). Under such conditions, the rogue nodes located in the network create a fake road accident scenario by broadcasting low–speed values in the beacon messages to all the vehicles in the region. The red region is where the rogue nodes are located, while the blue region is where the false messages are being broadcasted from the rogue nodes. Upon receiving the beacon messages from the rogue nodes, the vehicles start slowing down considering a road accident ahead. The green region is where the vehicles have not yet started braking

because they are not in the communication range of the rogue nodes. The F–RouND framework uses the Greenshields traffic flow model–based dynamic fog computing technique to detect the rogue nodes. However, in the case of an actual road accident scenario, the majority of the vehicles (i.e., honest nodes broadcast low–speed values in the beacon messages) can be easily identified and ignored.

2.3.3 Attack Model

Different types of attacks occur in VANETs. The F–RouND framework addresses the following false information attacks arising from beacon messages:

False information attack: Rogue nodes report an arbitrary event by modifying the values in the beacon messages with the vicious intent to cause damage to the network for their benefits. To create a greater impact on the network, the rogue nodes coordinate and collectively modify the speed values in the beacon messages at any time and broadcast low–speed values to the neighboring vehicles in the region to create an illusion of either a traffic congestion or a road accident ahead. Under extreme conditions, the false information messages propagated in VANETs may cause catastrophic consequences, including fatal vehicle collisions. The framework proposed herein effectively monitors the behavior of all the vehicles in the region to the detect rogue nodes, as discussed in Section 2.4.

2.4 Proposed F–RouND Framework

In this section, we discuss the working principle of the F–RouND framework. The F–RouND framework uses the emerging fog computing technique to detect the rogue nodes providing false information in specific low–speed values by altering the beacon messages in VANETs. The information about rogue nodes are then broadcasted to contain the damage. The proposed framework engages the dynamic fog computing

technique to detect the rogue nodes due to its unique characteristics, including low latency and high bandwidth. Moreover, the fog layer is at the proximity of users and performs computations at the network edge [13, 39].

Fog devices share their heterogeneous resources for computation and storage. Moreover, these devices can communicate and cooperate without the intervention of third parties. Fog devices can be either resource–constrained or resource–rich fog nodes with a powerful CPU, large memory, and storage. Individual OBUs are resource–constrained and cannot be used to process a large amount of data. Therefore, in our F–RouND framework, the OBUs of all the vehicles in the region are considered fog devices and are used for creating a dynamic fog layer at the proximity of the guard node. The dynamic fog layer is located at the network edge. It comprises fog nodes, which include gateways, fog devices, etc. The fog layer can be static at a fixed location or mobile on moving carriers, such as in the vehicular environment, and is responsible for processing information, such as beacon messages received from the neighboring vehicles, and temporarily storing it or broadcasting it over the network. One of the main advantages of the F–RouND framework is that unlike the existing frameworks [36-38, 43-55], the computation and storage power of the dynamic fog layer increase as the number of vehicles increases in the region, resulting in a lower data processing delay. This is due to the OBUs of all the vehicles being utilized to create the dynamic fog.

The F–RouND framework employs the guard node concept to detect all rogue nodes in the region. The vehicle with more neighboring vehicles in the communication range is considered as a guard node (Section 2.4.1). The guard node creates a dynamic fog layer using the OBUs of all the vehicles to compare and analyze beacon messages, specifically the vehicle speeds to detect rogue nodes. In addition to the vehicle speed, beacon messages exchanged between the vehicles also contain information, such as acceleration, braking status, and location. If there is a significant difference in vehicle

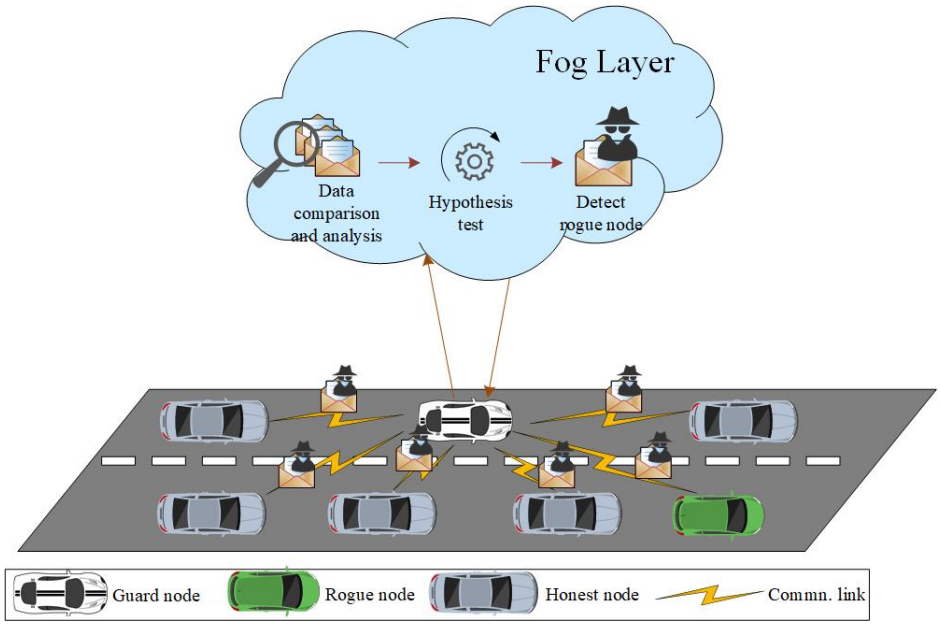


Figure 2.2: Execution scenario of the F–RouND framework in the presence of a rogue node using the dynamic fog computing technique.

speeds, the guard node classifies the vehicles as rogue nodes and the hypothesis test is performed to validate whether the rogue nodes are correctly identified.

If the hypothesis test yields speed values within the acceptance range, then the vehicles are considered as honest nodes; otherwise, the vehicles are highlighted as rogue nodes. Upon successful validation from the hypothesis test, the guard node broadcasts the information of rogue nodes to all the vehicles in the region. The vehicles start ignoring the subsequent beacon messages received from the rogue nodes to contain the damage. One such scenario of our framework is depicted in Fig. 2.2. The steps involved in the F–RouND framework in the rogue node detection are discussed in the subsections that follow.

2.4.1 Guard Node Selection

Rogue nodes are vehicles broadcasting low–speed values in beacon messages to change the normal behavior of the vehicles for their own benefit. In the F–RouND framework, the guard node analyzes the speed values in the beacon messages received from all

vehicles in the region to detect the rogue nodes. The following three assumptions are made in selecting the guard node. First, we assume that the guard node is the most trustworthy vehicle in the network. Thus, the guard node cannot turn out to be a malicious node in any given circumstances. Second, we assume that the center vehicle in the region acts as a guard node because the vehicle in the center has a higher number of neighboring vehicles in its communication range compared to the front and tail–end vehicles. Third, we assume that the total number of vehicles (N) in the region at any given time is at least two. The guard node needs at least two vehicles in the region to compare and analyze the beacon messages to detect the rogue nodes.

Most often, front–end vehicles are the rogue nodes (Fig. 2.1) and create an illusion of fake traffic congestion or road accident ahead to trap either one particular vehicle or all the vehicles traveling in the region and create catastrophic consequences, such as vehicle collisions. Moreover, as mentioned in Section 2.4.1, we assume that the guard node is the central vehicle in the region and cannot be malicious under any circumstances, thus ensuring that center vehicles are not the rogue nodes at any time interval.

Initially, we take the mean of the position vectors of all vehicles (i.e., P_1, P_2, ..., P_N) to find a unique center point ζ.

$$\zeta = \frac{1}{N} \sum_{i=1}^{N} P_i \tag{2.3}$$

We calculate the Euclidean distance between ζ and the position vector of each vehicle and determine the point with the minimum distance from the ζ. Finally, the vehicle located at this point is selected as the guard node, G_{veh}.

$$G_{veh} = \arg\min_{P_i \in X} \|\zeta - P_i\| \tag{2.4}$$

where, $X = \{P_1, P_2....,P_N\}$.

Sometimes, there may be an exceptional case where the point with the minimum distance may not be unique. In such a situation, our F–RouND algorithm will randomly select one vehicle among the minimum distance points as a guard node.

2.4.2 Cooperative Data Collection

The selected guard node (Section 2.4.1) collects the data from all the vehicles in the range. The V2V communication technique is used for broadcasting beacon messages. The vehicles share their information using the Greenshields traffic model discussed in Section 2.3.2. The vehicles receive beacon messages from all other vehicles in the region; thus, each vehicle knows about all the other vehicles in the region. However, to validate whether the received beacon messages are genuine, the guard node creates a dynamic fog layer by combining the computation resources (i.e., OBUs of all vehicles). In VANETs, the OBUs of individual vehicles are highly resource–constrained and cannot be used to analyze a large volume of data because it will result in a significant delay. Thus, a dynamic fog layer is used for processing the received beacon messages to detect the rogue nodes and to perform an extensive hypothesis test to validate whether the rogue nodes are correctly identified.

2.4.3 Message Format

In the F–RouND framework, each vehicle broadcasts beacon messages every 100 ms. The format of the beacon message of all the vehicles, except the guard node, is given below:

B_{msg} ($Speed$; $Position$; $Acceleration$; $Density$)

The guard vehicle modifies the existing beacon message format to include information about the rogue nodes in the region. Thus, apart from the usual parameters, the beacon messages from the guard node also include the following:

B_{msg} ($Speed$; $Position$; $Acceleration$; $Density$; Rlt; RID)

where, Rlt is the result of the hypothesis test and RID is the unique vehicle ID of the rogue nodes discussed in Section 2.4.5.

2.4.4 Speed and Density of Vehicles

As mentioned in Section 2.3.2, in the proposed framework (F–RouND), Greenshields mathematical model is used to model the traffic flow. The guard node calculates the density of vehicles in the region from the received beacon messages from the vehicles in the region and is given as follows:

$$\rho = B_{msg} \cdot N \tag{2.5}$$

where, B_{msg} is the beacon message broadcasted from one vehicle ID and N is the total number of vehicles in the region. The density window size is equal to the transmission and reception ranges of the vehicles in the network. In the F–RouND framework, each vehicle can transmit and receive messages up to 500 m (i.e., the vehicles can communicate with the neighboring vehicles up to 500 m ahead and behind them). Therefore, the communication window of each vehicle, including the guard vehicle, is 1000 m.

Table 2.1: Types of error and decisions in the null hypothesis testing

		Null hypothesis (H_0)	
		True	**False**
Null hypothesis (H_0) decision	**Accept**	No error	Type II error (False negative)
	Reject	Type I error (False positive)	No error

The speed and density of the vehicles are negatively correlated (Section 2.3.2); thus, the relationship between speed and density can be defined as follows:

$$S = S_{max} - \frac{\rho}{\rho_{max}} S_{max} \tag{2.6}$$

where, S_{max} is the speed of the vehicle when the density is zero and ρ_{max} is the maximum density, which is also the point at which the vehicle speed becomes zero.

The selected guard node (Section 2.4.1) creates a dynamic fog layer to compare and analyze speed values in the received beacon messages. If there is a significant speed difference in any of the beacon messages received, the guard node classifies the vehicle that broadcasted the corresponding speed value as a rogue node, and the hypothesis test (Section 2.4.5) is performed to validate whether the vehicle is rogue. Once the rogue nodes are identified, the guard node calculates the average density (ρ_{avg}) and the average speed (S_{avg}) to perform the hypothesis test.

$$\rho_{avg} = \frac{1}{N} \sum_{i=1}^{N} \rho_i \tag{2.7}$$

$$S_{avg} = \frac{1}{N} \sum_{i=1}^{N} S_i \tag{2.8}$$

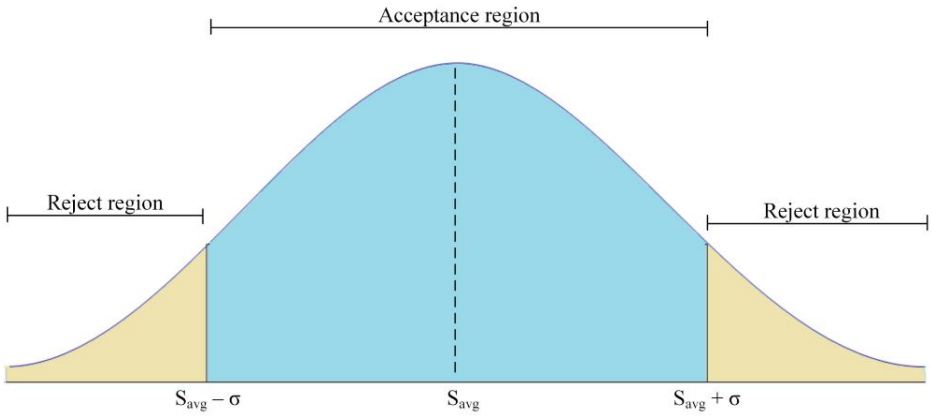


Figure 2.3: Hypothesis test of the F–RouND framework based on the average vehicle speed to determine the acceptance range values.

During the hypothesis test, the guard node compares the average speed (S_{avg}) with the individual speeds. The vehicles corresponding with the average speed are called honest nodes. In the case where the average speed difference is either high or low, the upper and lower bound values are calculated to decide whether a received speed should be accepted. The validation from the hypothesis test provides better performance of the F–RouND framework, resulting in a higher TPR and a lower FPR compared to the schemes presented in [36, 37, 38]. A brief explanation of the hypothesis test is illustrated in Section 2.4.5.

2.4.5 Hypothesis Test to Validate the Vehicle Speed

Hypothesis testing allows a confidence interval to be in a range of values that allows us to accept a claim with a certain confidence. Our framework performs a hypothesis test with the speed values received from all vehicles in the region, allowing the guard node to accept the speeds with a certain confidence. Moreover, hypothesis testing is a commonly used statistical technique when there are two different claims, of which only one claim can be true. In the F–RouND framework, except for the guard vehicle, we have two different claims for all the vehicles in the region (i.e., the vehicle is either honest or rogue). If the vehicle is honest, the guard node accepts the data; otherwise,

the vehicle is considered a rogue and the rogue node information is broadcasted over the region. We use the hypothesis test to validate the vehicle speed in the beacon messages, as presented in Fig. 2.3.

Two hypotheses are involved in the hypothesis testing approach: null hypothesis (H_0) and alternate hypothesis (H_a). The null hypothesis is the claim that must be tested, while the alternate hypothesis is everything else. If the null hypothesis is accepted, then the alternate hypothesis is rejected and vice versa. In the F–RouND framework, the null hypothesis (H_0) is that the speed value received is from an honest vehicle, while the alternate hypothesis (H_a) is that the speed value received is from a rogue node. Two types of error are associated with the hypothesis testing approach: the first type of error (Type I error) occurs when the null hypothesis is wrongly rejected, also known as a false positive; the second type of error (Type II error) occurs when the null hypothesis is wrongly not rejected, also known as a false negative. Table 2.1 presents the types of errors and decisions in the null hypothesis testing (H_0).

We use standard deviation (σ) to calculate the variation in average speed with the received speed values of all vehicles in the region as follows:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(S_{avg} - S_i)^2} \tag{2.9}$$

The speed values of the vehicles close to the average speed calculated using the guard node result in a low standard deviation, while the speed value of the vehicles that highly from the average speed value results in a high standard deviation. We calculated the confidence interval based on Eq. (9) to determine the acceptance range values. The upper and lower limits of the acceptance region are $S_{avg} - \sigma$ and $S_{avg} + \sigma$. The received speed values of the honest nodes always fall in the acceptance region when the null hypothesis is true (i.e., $S_{avg} - \sigma < S_{avg} < S_{avg} + \sigma$). Therefore, the guard node rejects speed values received outside the acceptance region

$(S_{avg} - \sigma > S_{avg} > S_{avg} + \sigma)$. The vehicles that broadcast false speed values are considered as rogue nodes.

Unlike the existing rogue node detection schemes [36,37,38], the F–RouND framework works efficiently for all vehicle densities and all road conditions. For example, in the case of a road accident or traffic congestion, the speed of all the vehicles dropping in the region will bring down the average speed; consequently, the speed values of all honest nodes remain in the acceptance region. The rogue nodes broadcasting false information can be easily detected depending on whether $S_{avg} > S_{avg} + \sigma$ or $S_{avg} < S_{avg} - \sigma$. Once the rogue nodes are identified, the guard node modifies the existing beacon message format in such a way that the rogue node ID and the result of the hypothesis (either 0 or 1) are embedded in it; then, the guard node broadcasts the information of rogue nodes to all the vehicles in the region.

$$Rlt = \begin{cases} 0; & S_{avg} - \sigma < S_{avg} < S_{avg} + \sigma \\ 1; & \text{Otherwise} \end{cases} \tag{2.10}$$

The beacon message from the guard node includes the following information: *(Rlt*, *RID*). All vehicles in the region start ignoring the beacon messages subsequently received from the rogue nodes to contain the damage.

2.4.6 Analysis of the Proposed F–RouND Framework

In this analysis, we calculated the probability of failure. System failure can occur due to loss of connectivity, insufficient capacity of fog, etc. The probability of system failure $P_{sysfail}$ is calculated as follows:

$$P_{sysfail} = \sum_{i=0}^{N,t_{max}} \binom{N, t_{max}}{i} d_f^i (1 - d_f)^{N,t_{max}{}^{-i}} \tag{2.11}$$

where, N is the number of vehicles, t_{max} is the maximum time taken by the vehicles to get connected, and d_f is the probability of success in the fog. A minimum number of failures leads to the maximum performance of the F–RouND framework.

2.4.7 F–RouND Rogue Node Detection Algorithm

Algorithm 1: : Rogue nodes detection algorithm
Input: G_{veh} receives B_{msg} from all vehicles in the region
Output: G_{veh} broadcasts the rogue node information to all the vehicles in the region

```
1  if (N ≥ 2) then
2  |  Calculate ζ ;
3  |  Calculate Euclidean distance ;
4  |  Assign G_veh ;
5  end
6  else
7  |  Goto step 24 ;
8  end
9  G_veh creates a dynamic fog layer from OBU's of all vehicles in the region ;
10 G_veh receives B_msg from all vehicles in the region ;
11 foreach each B_msg received do
12 |  Calculate ρ_avg ;
13 |  Calculate S_avg ;
14 |  Perform hypothesis test to validate each vehicle speed ;
15 |  if S in the acceptance range then
16 |  |  Declare the vehicle as honest node ;
17 |  end
18 |  else
19 |  |  Declare the vehcile as rogue node ;
20 |  |  Store the rogue node id ;
21 |  end
22 end
23 G_veh broadcasts rogue node information through B_msg ;
24 Terminate rogue node detection algorithm
```



Figure 2.4: Simulated maps of the F–RouND framework from the City of Norman, USA: (a) urban and (b) highway scenarios.

2.5 Performance Evaluation

This section evaluates the performance of our proposed framework discussed in Section 2.4. Each analysis is explained in the following subsections.

2.5.1 Simulation Setup

The main objective of our simulation was to evaluate the performance of the F–RouND framework in the presence of rogue nodes in both urban and highway scenarios (Section 2.4). We used OMNET++ and SUMO simulator to perform the simulations. SUMO is an open–source traffic simulator that provides a trace of vehicle movements, such as vehicle speed, position, and acceleration at the end of every simulation [57]. SUMO supports OpenStreetMap (OSM) to import real–world road networks, including buildings, water bodies, and traffic lights, for a realistic simulation. The output of the SUMO simulation was given as input to the OMNET++ simulator. OMNET++ is a discrete event simulator that provides a packet loss model, a node deployment model, a node mobility model, and a wireless signal propagation model to measure the network performance [58]. The node deployment and mobility models were used for determining the dynamic placement and movement of vehicles, respectively, while the

Table 2.2: Parameters used in the simulation of the F–RouND framework

Parameters	**Values**
Road length	6 km
Number of vehicles	500–4000
Number of lanes	2
Vehicle speed	30–70
Beacon message size	300 bytes
Transmission range	500 m
Simulation scenario	Urban and highway
Technique used	Fog computing
Protocol	IEEE802.11p
Simulator used	Omnet++, SUMO

wireless signal propagation and packet loss models were used for transmitting radio waves and measuring the number of packets dropped in the transmission, respectively.

To measure the performance of our F–RouND framework, we imported two maps from the city of Norman, United States of America. The first map represents the urban scenario, while the second shows the highway scenario of Norman, as represented in Fig. 2.4. The vehicles in the urban scenario had a lower mobility compared to those in the highway scenario as the vehicle speed in the urban scenario was limited to 30–45 mph whereas that in the highway scenario was up to 70 mph. The majority of the vehicles in the network were honest and broadcasted genuine messages to the other vehicles in the network. Thus, to assess the scalability and behavior of the F–RouND framework, we increased the presence of rogue nodes in the network up to 40%. Table 2.2 summarizes the most commonly used parameters used in the simulation.

All the vehicles in our simulations used the IEEE 1609 Wireless Access in Vehicular Environment (WAVE) protocol/DSRC stack, which builds on IEEE 802.1p WLAN standard and operates on seven reserved channels in the 5.9–GHz frequency band for vehicular communication. Among the seven channels, the DSRC stack has a Control Channel (CCH) responsible for broadcasting critical information, such as

beacon messages, and Six Service Channels (SCHs) for broadcasting noncritical information. The vehicles continuously adopted CCH to broadcast 300–byte beacon messages to the neighboring vehicles every 100 ms at an application rate of 3 Mbps and SCHs to randomly send 256 byte IP packets at an application rate of 6 Mbps. Our measurements were based on averaging the results obtained from 10 simulations. We varied the number of rogue nodes to be between 5% and 40% of the overall vehicles in the network. The total simulation time was 700 s. The rogue node detection process started immediately once all the vehicles entered the road. To provide stochasticity in the simulation, we used randomness in SUMO, which allows the vehicles to enter and exit from a random lane with a variable speed and destination. This eliminates a controlled environment with a predictable outcome, leading to a reality in the simulation.

OMNET++ uses a TCP–based client–server architecture, where OMNET++ acts as a client and SUMO acts as a server. It helped simulate the real streets of the city of Norman by considering the lanes, traffic lights, turns, and other traffic entities. In our F–RouND framework, speed values broadcasted in beacon messages are used to identify the rogue nodes in the region. Once the rogue node was detected in the simulation, the guard node changed the corresponding vehicle state in SUMO to a rogue node by sending a message to the OMNET++ client interface, which generated a set of commands and sent them to SUMO for execution, followed by broadcasting the rogue node information to all the vehicles in the region.

2.5.2 Performance Metrics

The simulations were performed based on the equations formulated in Section 2.4. The number of rogue nodes was increased from 5% to 40% to identify how successfully our proposed framework classifies trustworthy vehicles as honest nodes and malicious vehicles as rogue nodes. We considered data processing time, PLR, average

throughput, overhead, TPR, and FPR to evaluate the performance of the F–RouND framework and to compare our results with those of the Fog–IDS, IDS, and TEAM schemes:

- Data processing time: The time needed by the guard node to compare and analyze the beacon messages to detect rogue nodes in the region.
- PLR: The ratio of the number of lost packets to the total number of packets sent across a communication channel.
- Average throughput: Average rate of successfully broadcasted messages across a communication channel.
- Overhead: The overhead is the additional information exchanged between the vehicles to detect rogue nodes in the region.
- True positive rate: The percentage of rogue nodes is accurately detected and classified as rogue nodes.

$$\text{TPR} = \frac{\text{Number of rogue nodes detected correctly}}{\text{Total number of rogue nodes}} \tag{2.12}$$

- False positive rate: The percentage of honest nodes is incorrectly detected and classified as rogue nodes.

$$\text{FPR} = \frac{\text{Number of honest nodes detected incorrectly}}{\text{Total number of honest nodes}} \tag{2.13}$$

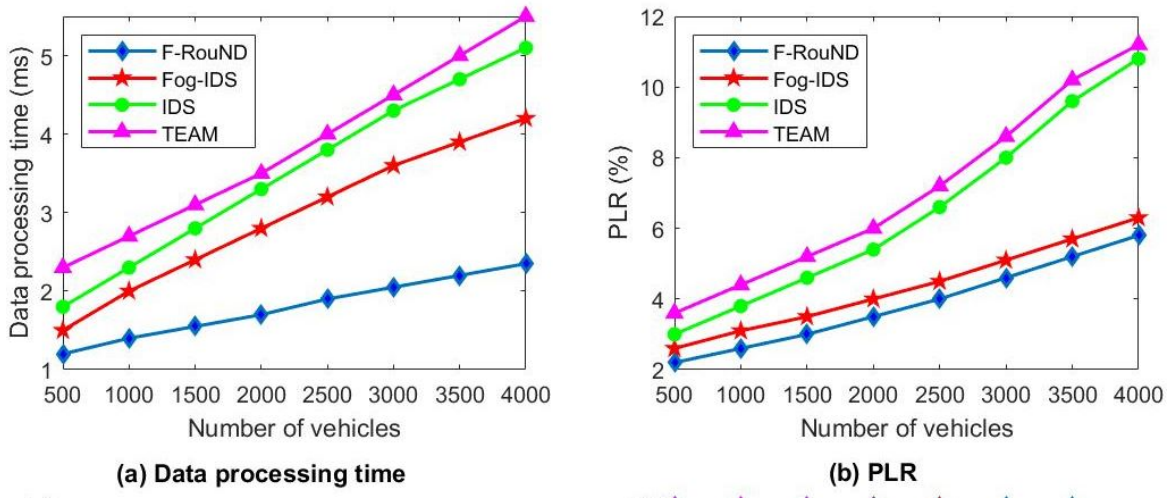


Figure 2.5: Comparison of urban scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (a) data processing time and (b) PLR.

2.6 Results

As mentioned in Section 2.5, we performed a simulation in two parts: the urban and highway scenarios. The vehicles in the urban scenario had a low mobility, while those in the highway scenario had a high mobility. During the simulation, the F–RouND framework was analyzed based on the equations formulated in Section 2.4 and the performance metrics illustrated in Section 2.5. The results are presented in the subsequent sections.

2.6.1 Urban Scenario

1) Data processing time: This is the time taken by the guard node to process and analyze the received beacon messages from the neighboring vehicles. When the number of vehicles increased from 500 to 4000, the data processing time increased as the guard node needed to process a large number of beacon messages received from all the vehicles in the region. However, as mentioned in Section 2.4, the computation power of the guard node increased when the number of vehicles increased in the region because the OBUs of all vehicles utilized in creating the dynamic fog layer resulted

in a 45% lower processing delay at all vehicles densities compared to the schemes presented in [36, 37, 38]. In the 4000–vehicle simulation, the data processing time was 43%, 52%, and 57% lower than that in the Fog–IDS, IDS, and TEAM schemes, respectively as shown in Fig. 2.5a. The results show that our F–RouND framework is efficient, scalable, and can handle high vehicle densities.

2) PLR: The PLR increased when the number of vehicles increased in the region because an increase in the number of vehicles receives a large number of beacon messages from the neighboring vehicle. Thus, the load on the dynamic fog layer increased, which consequently resulted in a packet drop when the load reached the maximum capacity of the dynamic fog. However, the high computation power of the fog resulted in an optimum network capacity even at high vehicle densities in an urban scenario. The PLR was calculated for a number of vehicles ranging from 500 to 4000, as shown in Fig. 2.5b. In the 4000–vehicle simulation, the PLR was 8%, 45%, and 47% lower than that in the Fog–IDS, IDS, and TEAM schemes, respectively.

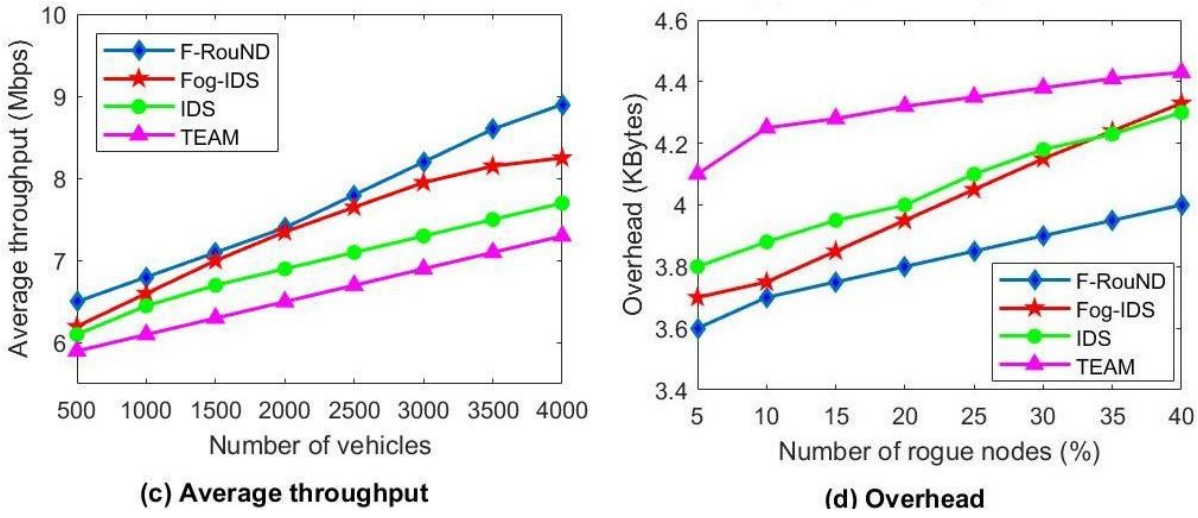


Figure 2.5: Comparison of urban scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (c) average throughput and (d) overhead.

3) Average throughput: In our F–RouND framework, the average throughput increased when the number of vehicles increased from 500 to 4000 because a large

number of beacon messages were successfully broadcasted to all vehicles in the region. This was due to the high scalability of our dynamic fog, low data processing delays (Fig. 2.5a), and low PLR (Fig. 2.5b) at high vehicle densities. In the 4000–vehicle simulation, the average throughput in an urban scenario was 9%, 17%, and 23% higher than that in the Fog–IDS, IDS, and TEAM schemes, respectively. The evaluation of average throughput shows the robustness and efficiency of the F–RouND framework, as shown in Fig. 2.5c.

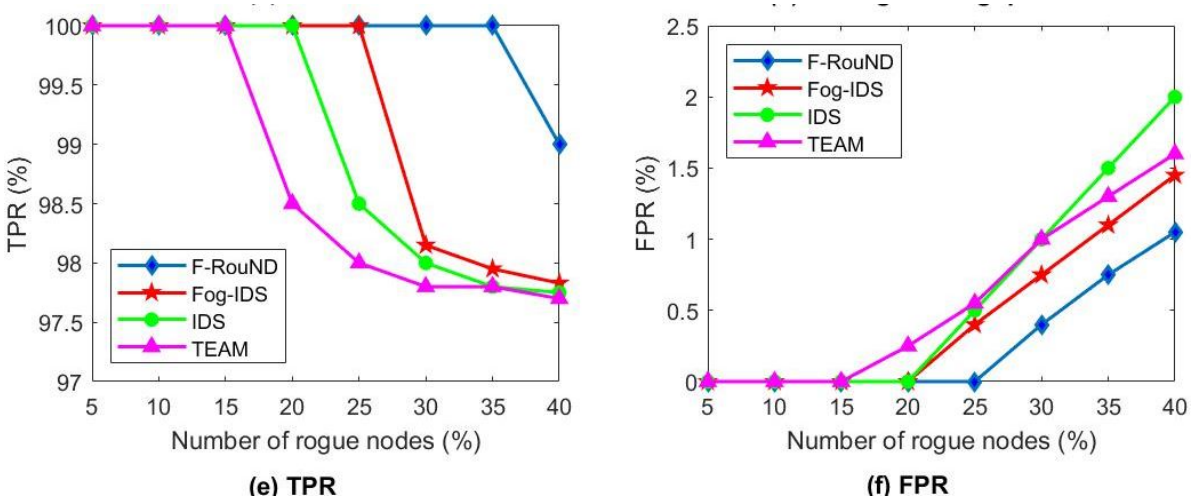


Figure 2.5: Comparison of urban scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (e) TPR and (f) FPR.

4) Overhead: The overhead of the F–RouND framework was calculated against the number of rogue nodes as shown in Fig. 2.5d, and increased at all vehicle densities due to an extensive hypothesis test needed to validate whether or not the rogue nodes were correctly identified (Section 2.4). However, unlike existing approaches [36,37,38], our F–RouND framework does not require any additional information, such as past vehicle data, trust score, and digital signature exchanged between the guard node, to detect rogue nodes in the region resulting in 12% lower overhead in an urban scenario even when the number of rogue nodes increased up to 40% as shown in Fig. 2.5d. For example, when the number of rogue nodes was 30% in the region, the overhead was 10%, 9%, and 13% lower than that of the Fog–IDS, IDS, and TEAM schemes,

respectively.

5) TPR: TPR was calculated against the number of rogue nodes, as shown in Fig. 2.5e. When the number of rogue nodes was more than 35%, the TPR marginally decreased to 99%. It was difficult to detect the rogue node when the speed variation was gradual. However, the target rogue node suddenly decreased the speed values to generate catastrophic consequences like vehicle collisions. Thus, the F–RouND framework can detect the rogue nodes even at high vehicle densities, resulting in a higher TPR compared to [36, 37, 38].

6) FPR: The increase in the FPR increased the rogue nodes in the region and deteriorated the performance of the rogue node detection schemes. One of the main advantages of the F–RouND framework is that the rogue node detection relies only on the speed values in the beacon messages broadcasted by all vehicles in the region without considering any trust scores or past vehicles, resulting in 36% lower FPR even at 40% rogue nodes in the region compared to [36, 37, 38] schemes. For example, when the number of rogue nodes was 40% in the region, the FPR was 31%, 51%, and 38% lower than that of the Fog–IDS, IDS, and TEAM schemes, respectively as shown in Fig. 2.5f.

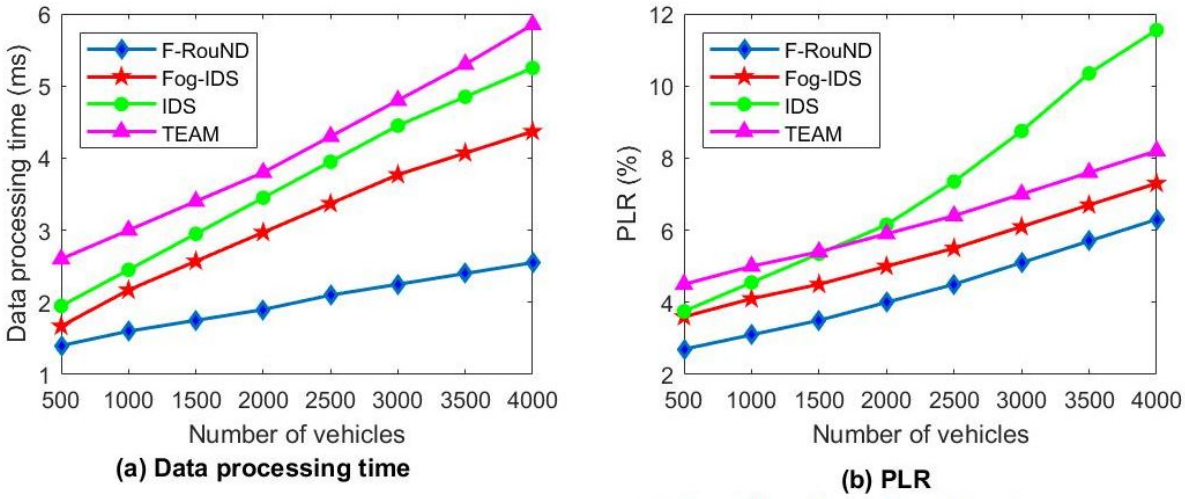


Figure 2.6: Comparison of highway scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (a) data processing time and (b) PLR.

2.6.2 Highway Scenario

1) Data processing time: Fig. 2.6a shows the data processing delay for the number of vehicles ranging from 500 to 4000 in a highway scenario. The F–RouND framework showed a relative insensitivity to vehicle counts because the guard node adopted a dynamic fog layer for processing the beacon messages received from neighboring vehicles (Section 2.4). Therefore, irrespective of the simulation scenario (i.e., either urban or highway scenario), the delay remained similar. The data processing delay at the highway scenario was 44% lower compared to those in [36, 37, 38], resulting in a highly robust and efficient framework. For example, in the 3000–vehicle simulation, the data processing delay was 40% lower than that in the Fog–IDS, IDS, and TEAM schemes.

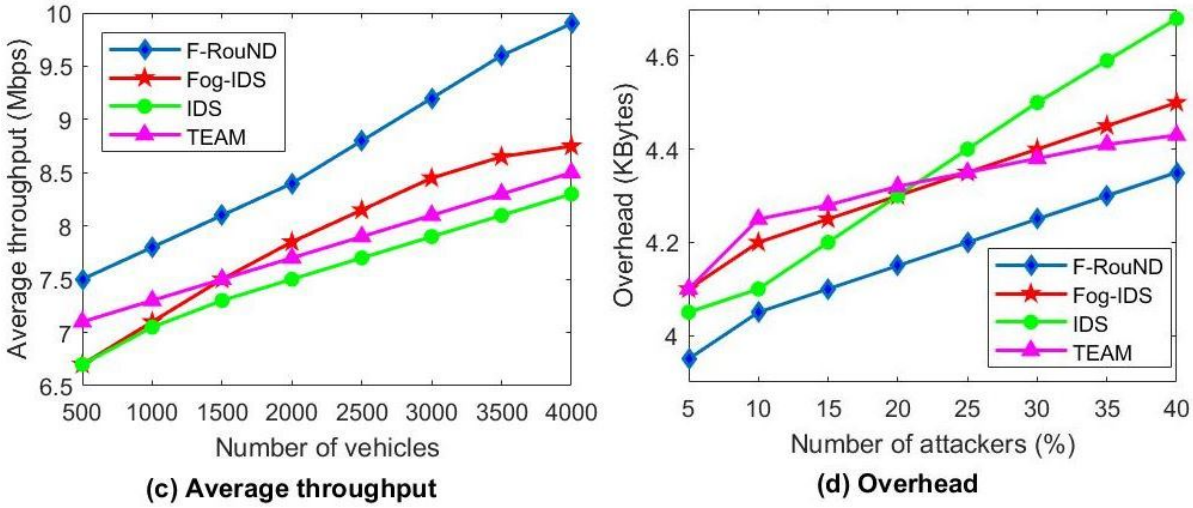


Figure 2.6: Comparison of highway scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (c) average throughput and (d) overhead.

2) PLR: The PLR of our F–RouND framework in a highway scenario was marginally higher compared to the urban scenario was due to the high mobility of the vehicles, which resulted in a collision of some packets. However, the PLR of our highway scenario was lower compared to that of the schemes presented [36, 37, 38] at all vehicle densities. The PLR was calculated against the number of vehicles and increased for

all schemes with increasing number of vehicles. In the 4000–vehicle simulation, PLR was 12%, 41%, and 23% lower than that in the Fog–IDS, IDS, and TEAM schemes respectively, as shown in Fig. 2.6b.

3) Average throughput: Due to a large number of messages being successfully broadcasted to the neighboring vehicles in the dynamic fog region, the average throughput of the F–RouND framework was higher compared to that in the schemes presented in [36,37,38] in the highway scenario. The average throughput was calculated against the number of vehicles and increased when the number of vehicles increased from 500 to 4000 as shown in Fig. 2.6c. In the 4000–vehicle simulation, the average throughput was 13%, 19%, and 16% higher than that in the Fog–IDS, IDS, and TEAM schemes, respectively.

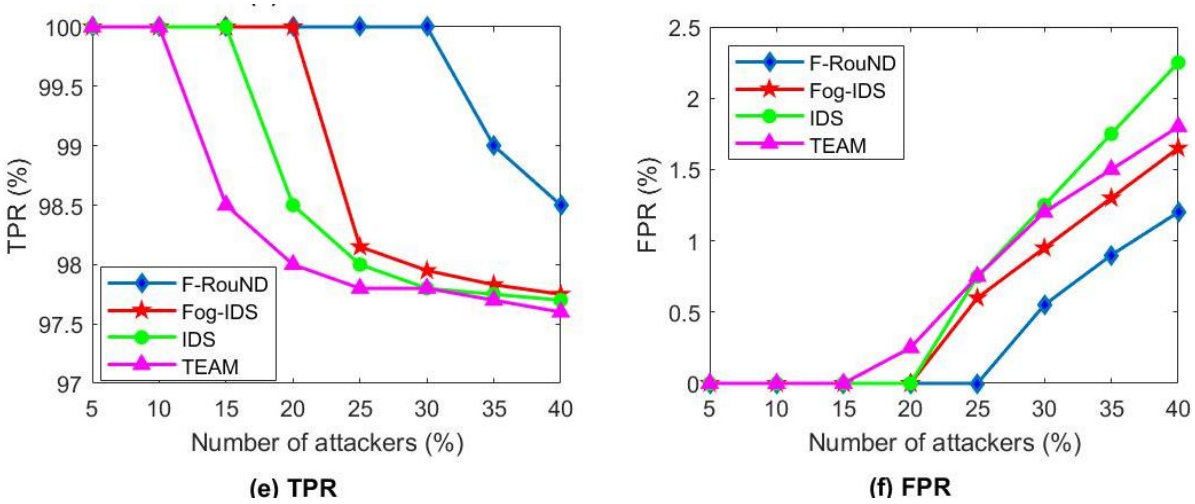


Figure 2.6: Comparison of highway scenarios of the F–RouND framework with Fog–IDS, IDS, and TEAM schemes: (e) TPR and (f) FPR.

4) Overhead: Fig. 2.6d shows the effect of change in the number of rogue nodes on the overhead in a highway scenario. As mentioned in Section 2.6.1, the validation of the hypothesis test increased the overhead of our framework when the number of rogue nodes increased in the region. However, the F–RouND framework only relies on speed values in the received beacon messages when detecting rogue nodes. Thus, the overhead of the F–RouND framework in the highway scenario was 10% lower

compared to that in the schemes presented in [36, 37, 38]. For a network with 40% rogue nodes, the overhead was 7%, 13%, and 6% lower than that in the Fog–IDS, IDS, and TEAM schemes, respectively.

5) TPR: The TPR of the F–RouND framework in the highway scenario identified the rogue nodes correctly up to 30% rogue nodes in the region and slightly decreased to 98.5% when the number of rogue nodes increased to 40% as shown in Fig. 2.6e. It was difficult to detect the rogue nodes broadcasting false information when the speed varied gradually in the received beacon messages. However, to generate either a road accident scenario, the rogue node suddenly decreased the speed values, resulting in a higher TPR compared to that in the Fog–IDS, IDS, and TEAM schemes at all vehicle densities.

6) FPR: An increase in FPR is a critical issue for any rogue node detection schemes as it increases the number of rogue nodes that may lead to severe network damage. The twofold process of the F–RouND framework (i.e., comparison of the speed values in beacon messages) for detecting rogue nodes and validating an extensive hypothesis test to determine whether the rogue nodes are correctly identified or not resulted in 32% lower FPR compared to the [36, 37, 38] schemes at all vehicle densities as shown in Fig. 2.6f. For a network with 40% rogue nodes, the FPR was 25%, 46%, and 32% lower than Fog–IDS, IDS, and TEAM schemes, respectively.

2.7 Summary

We studied herein the challenges in the existing rogue node detection schemes [36, 37, 38], including high delay, poor resource utilization, high FPR, and low TPR at high vehicle densities when the number of rogue nodes increases in a region. To address these limitations and provide an efficient scheme to detect the false messages broadcasted over the network, we proposed Greenshield's traffic flow–based fog comput-

ing technique called F–RouND for rogue node detection. The F–RouND framework demonstrated the effectiveness of the fog computing technique in determining rogue nodes, even when the number of rogue nodes increases by up to 40% in the region.

The simulations were performed based on metrics such as the data processing time, PLR, average throughput, network overhead, TPR, and FPR to evaluate the performance of the proposed framework using OMNET++ and SUMO simulator. Results showed that the F–RouND framework is scalable, efficient, robust, and performs up to 38% better than the schemes presented in [36, 37, 38]. Moreover, the performance of our extensive hypothesis test in validating the rogue nodes ensured 45% lower processing delay, 36% lower FPR, and 12% lower overhead at the urban scenario and 44% lower processing delay, 32% lower FPR, and 10% lower overhead at the highway scenario compared to that in the Fog–IDS, IDS, and TEAM techniques [36, 37, 38]. The F–RouND framework does not depend on any roadside infrastructures such as RSUs or trust scores or past vehicle data in rogue node detection, which is a major advantage compared to the existing rogue node detection schemes.

Chapter 3

Fog-based Rogue Nodes Detection in VANETs: Sybil Attack

3.1 Introduction

A Sybil attack uses a single node to simultaneously operate many active fake identities (or Sybil identities) within a peer-to-peer network [3, 59]. The term Sybil node describes a non-existent node rogue node claiming its presence. This type of attack aims to undermine the authority or power in a reputable system by gaining the majority of influence in the network. The Sybil attack becomes more severe as the number of rogue nodes increases in the region. Rogue nodes may cause a Sybil attack to create an illusion of traffic congestion in order to reroute other honest vehicles in the network to trap it or to cause catastrophic consequences, such as the collision of vehicles. Therefore, efficient detection of the rogue nodes causing a Sybil attack is crucial in containing network damage and establishing a secure VANET environment. Speed and position verification is one of the promising approaches to detect the Sybil attack [60].

Previous authors used either cryptography, trust scores, or past vehicle data to detect rogue nodes invlolved in Sybil attack. Baza et al. [61] presented an intrusion IDS to detect rogue nodes based on the power of works (PoW) algorithm. Each vehicle starts the trajectory by transmitting the vehicle data using the public key and the signature using the private key to the RSUs. The RSUs verify the trajectory of the vehicles based on the historical vehicle data. If multiple trajectories for the same vehicle exists, the corresponding vehicle will be considered a rogue, and

the information of the rogue node will be broadcasted to all vehicles in the region. The proposed scheme [61] suffers from a high delay and overhead in encrypting and decrypting the vehicle data. Zaidi et al. [37] proposed an IDS, where each vehicle utilizes its OBU to detect false data propagated by the rogue nodes based on the past vehicle data. Ayaida et al. [62] proposed a trust model (TM) to detect false messages in VANETs based on the correctness of the data broadcasted in beacon messages. However, [37, 62] cannot be used in high vehicle dense regions, such as Manhattan and other downtown regions, due to high PLR and FPR.

FSDV exploits statistical approach phenomena to generate a residual corresponding to the difference between the actual speed of the guard node and the received speed of the vehicles in a distributed way by using the received beacon messages. A significant deviation of this residual from a dynamic threshold is considered a potential indicator of the Sybil attack. Thus, the guard node incorporates a beamforming technique that sends a challenge packet to the vehicleś claimed location. If the vehicle is at the claimed position, it should receive and send back a challenge packet. Failure to obtain the response packet confirms the suspicion of a Sybil attack.

The selection of the dynamic threshold also directly impacts the Sybil attack detection probability. Thus, the threshold has to be chosen properly in order to well detect and avoid false-positive detections. We adopt fog computing, as it offers low latency, low network overhead, and high bandwidth compared to traditional communication techniques.

The *novelty* of our proposed work lies in providing low data processing delays and FPR at high vehicle densities. In addition, FSDV does not depend on any roadside infrastructures like RSUs or trust scores or past vehicle data in rogue nodes detection. The *difference* between our framework and existing schemes [37,61,62] is, each vehicle uses its OBU or RSU to detect rogue nodes. RSUs are not uniquely deployed in all VANETs regions. Moreover, the overloaded RSUs yields high data processing delay

and FPR. OBUs of an individual vehicle is highly resource-constrained encounters a high delay in analyzing the data at high vehicle densities. Whereas, in the FSDV, the guard node combines OBUs of all vehicles in the region in creating the dynamic fog. Utilizing OBUs of all vehicles increases the computational power of dynamic fog resulting in low data processing delay and FPR.

Our *objective* is to reduce the latency in detecting rogue nodes and decrease FPR at high vehicle densities. We considered three existing rogue nodes detection schemes for comparison: PoW [61], IDS [37], and TM [62]. The performance of FSDV was carried out using OMNET++ and SUMO simulators with up to 4000 vehicles and 40% rogue nodes. Our results lead to an exciting conclusion that our framework reduces the latency and FPR, and performs up to 32% better than the existing Sybil attack detection schemes [37, 61, 62].

The *contributions* of this chapter are as follows:

1. We proposed a framework, FSDV, that uses statistical techniques to detect the Sybil attack in VANETs with low delay and low FPR.

2. We introduced the guard node in FSDV, which uses an OBU-based fog computing technique to compare speed values and position values received in the beacon messages from the vehicles in the region.

3. We performed an extensive simulation by creating a dynamic fog layer under varying vehicular and network conditions to determine false information broadcasted by rogue nodes.

3.2 Related Work

This section presents an overview of the most recent existing schemes that detect Sybil attacks in VANETs. Yu et al. [63] and Feng et al. [64] proposed a trust-based

scheme to detect rogue nodes in VANETs. Based on the correctness of the data in beacon messages, positive or negative trust values are assigned. Positive and negative trust values represent the normal and abnormal behavior of the vehicles, respectively. When the calculated trust of any vehicle reaches a predefined threshold limit is known as a rogue node and then the information is broadcasted to all the vehicles in the region.

Iwendi et al. [65] proposed a spider money technique, which utilizes the RSUs in the region to authenticate the private key associated with beacon messages to identify malicious data broadcasted by the rogue nodes. Once the rogue nodes are identified, the RSUs send rogue nodes information to the department of motor vehicles (DMV) to withdraw the vehicle's permit to reduce the potential damage to the network. The frameworks [63, 64, 65] encounter a low TPR and high FPR in detecting rogue nodes.

Benadla et al. [66] presented a cryptography-based IDS using fog computing. The proposed scheme considers RSUs as fog nodes for rogue nodes detection. However, the approach [66] encounters a high processing delay and overhead in detecting rogue nodes when the RSUs are overloaded or not available in the region. Concone et al. [67] proposed a machine learning algorithm to classify whether received data is valid or not based on the historical vehicle data. The results of the classification are then combined to detect rogue nodes broadcasted false information. However, this approach [67] encounters high delay and overhead at high vehicle densities.

Quevdo et al. [68] illustrated a cryptographic–based message authentication scheme to detect false messages propagated among vehicles in VANETs. The proposed framework [68] employs a ring signature to generate the public and private keys for each vehicle in the network. When the sender broadcasts a beacon message to the neighboring vehicles, a ring signature gets generated and transmitted along with the message. The receiver accepts this message upon successful verification, which helps identify whether the message is modified in the middle. However, a significant delay is asso-

ciated with the generation of the private and public keys for each vehicle when the RSUs are overloaded or not available in the region. Rabieh et al. [69] proposed a machine learning algorithm that verifies whether the data received from the vehicles in the region are valid by comparing them with the past vehicle data using classification and decision tree techniques. This approach [69] suffers from low TPR and high FPR when the number of vehicles increases in the region.

Pattanayak et al. [70] and Sagi et al. [71] proposed trust–based intrusion detection frameworks for rogue node detection. The models proposed in [70, 71] assigns a trust value to all the vehicles in the region based on their behavior. The vehicles that either drop or alter the message are considered rogue nodes with negative trust values. Vehicles other than the rogue nodes are trustworthy vehicles with positive trust values. The trust values of the vehicles are maintained in the score table, which gets updated whenever messages are broadcasted by the vehicles. Each vehicle has a copy of the score table containing information about all the vehicles in the region. Thus, the messages received from the rogue nodes are ignored to contain network damage. As such, the frameworks proposed in [70,71] suffer from high PLR and FPR in detecting the rogue nodes.

To overcome the limitations of the existing Sybil attack detection schemes [37, 59-71], we propose the FSDV framework, which utilizes only vehicle speed values in beacon messages and does not depend on either trust score, cryptography, or past vehicle data in rogue nodes detection.

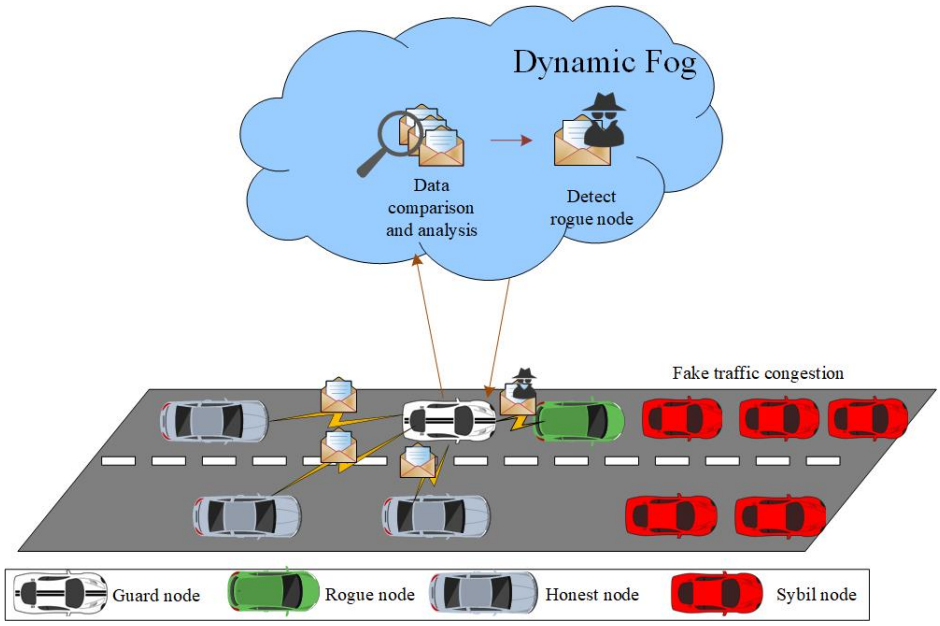


Figure 3.1: Execution scenario of FSDV in the presence of a rogue node using fog computing technique.

3.3 Proposed FSDV Framework

The FSDV framework engages fog computing in detecting rogue nodes involved in the Sybil attack. The rogue nodes broadcast multiple messages with different vehicle IDs to all vehicles in the region, thereby giving a false impression of fake traffic congestion ahead by lowering the speed values and false position values in the beacon messages [72, 73, 74]. One such scenario is illustrated in Fig. 3.1. In this chapter, we employ the concept of guard node to detect the Sybil attack in VANETs. The vehicle which has more neighboring vehicles in its communication range will act as a guard node. Guard node dynamically creates a fog utilizing OBUs of all vehicles to detect rogue nodes broadcasting lower speed values in the region. Once the fog layer is deployed, the guard node starts adding the neighboring vehicles to the list of neighbors, as met for the first time. Otherwise, it updates the timestamp of the neighboring vehicles. The neighbors list is updated continuously when the beacon messages are broadcasted in the region (i.e., for every 100ms).

The detection of rogue nodes is achieved by analyzing the speed, and position values broadcasted in beacon messages. The difference between the speed of the guard node and the received speed value from the neighboring vehicles, which is denoted as $(S - S_{rcvd})$ is compared with the dynamic threshold (S_{th}), which depends on the road condition and traffic flow model. If the speed difference is greater than the dynamic threshold, a potential Sybil attack is detected. This is because all the vehicles in the region broadcast similar speed values as they are in similar traffic conditions and dependent on other vehicles. Otherwise, the vehicles are highlighted as honest nodes.

If the guard node suspects the Sybil attack, it sends a challenge packet to the vehicleś claimed position using the beamforming technique. The guard node does not send a challenge packet to the vehicle's location; instead, it estimates the location of the vehicle at the next time interval based on the speed received in the beacon message and propagation delay. If the vehicle is at the claimed location, it should be able to receive the packet and send it back to the guard node. After sending the challenge packet, the guard node sets a timer and waits for the response. If the timer expires without receiving the packet, the guard node declares the vehicles as Sybil nodes.

If the suspected vehicle is one of the rogue nodes, the response packet will be received as it is in the claimed location; however, the speed of the rogue node does not correspond with the speed of other vehicles in the network. The guard node broadcasts the information of rogue and Sybil nodes to all vehicles in the region to ignore the beacon messages received further to contain the network damage. The adoption of fog computing increases the computation power of the guard node as the OBUs of all vehicles are utilized in creating a dynamic fog, resulting in lower data processing delay compared to [37, 61, 62] schemes.

3.3.1 Selection of Guard Node

Rogue nodes are the vehicles broadcasting low-speed values and false position to change the normal behavior of the vehicles for own benefits [75, 76, 77]. In the FSDV framework, the guard node analyzes the speed values in the beacon messages received from all vehicles and position values of suspected vehicles in the region to detect rogue nodes. The following two assumptions are made in the selection of the guard node: First, we assume the center vehicle in the region will act as a guard node (G_{veh}), as the vehicle in the center has more number of the neighboring vehicles in its communication range compared to the front and tail-end vehicles. Second, we assume that the total number of vehicles (N) in the region at any given time is at least two. The guard node needs at least two vehicles in the region to compare and analyze the beacon messages to detect rogue nodes.

Initially, we take the mean of position vectors of all vehicles (i.e., P_1, P_2, ..., P_N) to find a unique center point ζ.

$$\zeta = \frac{1}{N} \sum_{i=1}^{N} P_i \tag{3.1}$$

We calculate Euclidean distance between ζ and the position vector of each vehicle and then determine the point that has the minimum distance from the ζ. Finally, the vehicle located at this point will be selected as the guard node, G_{veh}.

$$G_{veh} = \arg \min_{P_i \in X} \|\zeta - P_i\| \tag{3.2}$$

Where, $X = \{P_1, P_2....,P_N\}$.

3.3.2 Density and Speed of Vehicles

In the FSDV framework, Greenshield's traffic flow model is used to calculate the speed of the guard node and the density of the vehicles in the region. Greenshield traffic model works under the assumption of density (ρ), and the speed of the vehicles (S) is negatively correlated. The density can be calculated as:

$$\rho = B_{msg} \cdot N \tag{3.3}$$

Where, B_{msg} is the beacon message received from one vehicle id and N is the total number of vehicles in the region. The relationship between speed and density, also the speed of the guard node can be defined as:

$$S = S_{max} - \frac{\rho}{\rho_{max}} S_{max} \tag{3.4}$$

Where S_{max} is the speed of the vehicle when density is zero and ρ_{max} is the maximum density, also point at which speed of the vehicles becomes zero. The calculated speed value of the guard node (S) is compared with the received speed values of the neighboring vehicles (S_{rcvd}). If the difference in speed values is greater than the dynamic threshold (S_{th}), i.e., $S - S_{rcvd} > S_{th}$, position verification is carried out on suspicious vehicles to determine whether or not Sybil attack takes place in the region. The vehicles that broadcasted malicious information but were still present in the claimed location are considered rogue nodes, and those that broadcasted malicious information not present in the claimed location are considered Sybil nodes. Finally, the guard vehicle broadcasts the information of rogue and Sybil nodes to all the vehicles in the region.

3.3.3 FSDV Algorithm

Algorithm 2: FSDV Sybil Attack Detection algorithm
Input: G_{veh} receives B_{msg} from all vehicles in the region
Output: G_{veh} broadcasts information of rogue and Sybil nodes

if *($N \geq 2$)* **then**
  Calculate ζ and Euclidean distance ;
  Assign G_v ;
end
G_{veh} dynamically creates a fog ;
G_{veh} receives B_{msg} from all vehicles in the region ;
else
  Terminate the algorithm
end
foreach *each B_{msg} received* **do**
  if *($B_{msg}.Sender \notin Neighbors_{list}$)* **then**
    $Neighbors_{list}$.add ($Sender$)
  end
  $Neighbors_{list}[Sender].T_{stamp} = B_{msg}.T_{stamp}$;
  Calculate S ;
  if *($S - S_{rcvd} < S_{th}$)* **then**
    Declare the vehicle as honest node ;
  end
  else
    Send challenge packet to suspected vehicles ;
    if *(Suspected vehicle $\notin$ claimed location)* **then**
      Declare the vehicle as sybil node;
    end
    else
      Declare the vehicle as rogue node ;
    end
    Store the rogue and guard node ids;
  end
end
G_{veh} broadcasts rogue node information and terminate the algorithm

3.4 Mathematical Model Analysis

To validate the performance of the proposed approach, discussed in Section 3.3, we investigate in this section the theoretical expression for the delay and the probability of detecting the rogue nodes correctly in the Sybil attack.

3.4.1 Analysis of Delay in FSDV

As our objective is to reduce the latency significantly in detecting rogue nodes compared to [37, 61, 62] schemes, we have analyzed different types of delays, such as communication delay, queuing delay, and data processing delay associated with the FSDV framework, illustrated in Section 3.3. The communication delay, D_c associated with the guard node in transmitting the received beacon messages from all vehicles in the region to the fog layer is given by:

$$D_c = \frac{x}{r} = \frac{x}{b \log_2 \left(1 + \frac{tc^2}{\sigma^2}\right)} \tag{3.5}$$

Where, x is the number of bits in the beacon messages, t is the transmission power of fog devices at the fog layer, c is the channel coefficient between devices the guard node and the fog layer, σ^2 is the power of white Gaussian noise, r is the transmission rate calculated from Shannon channel capacity theorem, and b is the bandwidth of the link established between the guard node and the fog layer.

Devices associated with the fog layer use the M/M/1 queuing model for preparing the beacon messages to be computed, where the arrival and service time are represented as $\frac{1}{\lambda_1}$ and $\frac{1}{\mu_1}$, respectively. The queuing delay (D_q) associated with a fog layer of our FSDV framework is given by:

$$D_q = 1 - \frac{1}{\lambda_1} + \frac{{\lambda_1}^2}{\mu_1^2(\mu_1 - \lambda_1)} \tag{3.6}$$

The data processing delay (D_p) indicates the required time to process all received beacon messages in the fog layer to detect rogue nodes is given by:

$$D_p = \frac{TC(x)f}{c} \tag{3.7}$$

Where c is the computation capability of the fog layer, f is the number of CPU cycles required for computing one bit of data at the fog layer, and $TC(x)$ is the time complexity for x bits of data in the beacon messages, depends on the algorithm discussed in Section 3.3. The total delay (D_t) of our FSDV framework in the fog layer is given by:

$$D_t = D_c + D_q + D_p \tag{3.8}$$

3.4.2 Malicious Nodes Verification

Assume the beacon messages from all other vehicles in the region are successfully received by the guard vehicle. The probability of analyzing the speed and position values in the beacon messages to detect the rogue nodes correctly is given by:

$$X(P_1P + P_2P) \tag{3.9}$$

The parameter P is the probability that beacon messages positively reach the guard vehicle from all other vehicles in the region. P_1 represents the probability of predicting the honest vehicle correctly if the difference between the calculated speed of the guard node and the honest vehicle is less than the dynamic threshold (i.e., S – $S_{rcvd} < S_{th}$). P_2 represents the probability of predicting the malicious node (either rogue node or Sybil node) correctly if the difference between the calculated speed of the guard node and the rogue node is greater than the dynamic threshold (i.e., S – $S_{rcvd} > S_{th}$). X represents the probability that the guard node successfully creates a dynamic fog layer to analyze the speed and position values in the beacon message to detect the rogue and Sybil nodes in the region. The probability of incorrectly predicting the malicious nodes, P_i is given by:

$$\begin{aligned} P_i &= 1 - [XP_1P + XP_2P] \\ &= 1 - [XP_1 + XP_2]P \\ &\approx 1 - XP_2P \end{aligned} \tag{3.10}$$

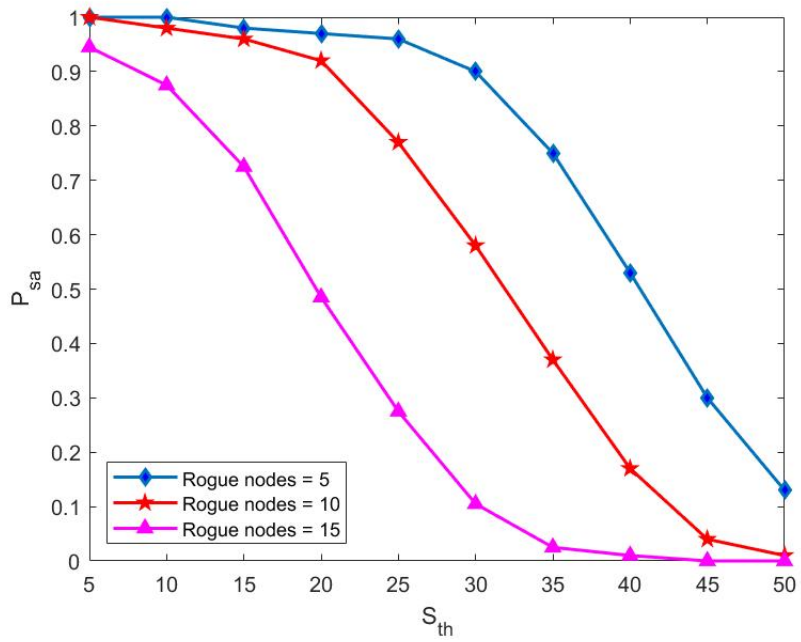


Figure 3.2: Effect of dynamic speed threshold in the Sybil attack detection probability.

3.5 Performance Evaluation

This section evaluates and analyzes the performance of the FSDV framework discussed in Section 3.3.

3.5.1 Evaluation of Speed Threshold in FSDV

As discussed in Section 3.3, a significant deviation of the speed values from a dynamic threshold (S_{th}) is considered a potential Sybil attack [78, 79]. The threshold value is more dynamic as it depends on the speed of the vehicles in the region. Thus, when the speed of the vehicle decreases, the dynamic threshold value also decreases in the region. For example, the dynamic threshold value in the dense vehicle region, such as the downtown region is lower, compared to the dynamic threshold value in the fluid traffic region. This is due to the speed of the vehicles in a high dense vehicle region is lower compared to the fluid traffic region.The Fig. 3.2 shows how the selected speed threshold S_{th} impacts the detection probability P_{sa}.

Table 3.1: Parameters used in Simulation of the FSDV Framework

Parameters	**Values**
Number of vehicles	500-4000
Road length	5 Miles
Number of lanes	2
Vehicle speed	30-65 Miles/hr
Transmission range	500 m
Beacon message size	256 bytes
Protocol	IEEE802.11p

From Fig. 4.2, we can infer that the probability of Sybil attack detection is high when the speed threshold (S_{th}) is low. This is due to we were able to identify all the rogue nodes in the region. However, the low-speed threshold may hide false positive detection. When the speed threshold (S_{th}) increases, the detection probability decreases since we can miss some rogue nodes due to the high-speed threshold. Therefore, to detect all the rogue nodes in the region and to reduce FPR, the speed threshold has to be chosen dynamically based on the speed of the vehicles in the region.

3.5.2 Simulation Setup

The simulations are carried out using OMNET++ and SUMO simulators. SUMO is an open-source traffic-events simulator, provides a trace of vehicle movements, such as vehicle speed, position, acceleration, etc. at the end of every simulation for a map imported from OpenStreetMap [80, 81, 82]. To perform the simulation, we imported the map of the city of Norman, Oklahoma into the SUMO simulator. The output of the SUMO simulator, i.e., the trace of vehicles, is given as input to the OMNET++ simulator for rogue nodes detection. To assess the scalability and behavior of the FSDV framework, we increase the number of vehicles up to 4000 and the presence of rogue nodes in the network up to 40%. Table 3.1 summarizes the most commonly used parameters used in the simulation.

3.5.3 Performance Metrics

The simulations were performed based on the equations formulated in Section 3.3 and 3.4. We considered the following metrics to evaluate the performance of our FSDV framework and to compare our results with PoW, IDS, and TM schemes:

- Data processing time: The time needed by the guard node to analyze the beacon messages received in the region.
- PLR: The ratio of the number of lost packets to the total number of packets sent across a communication channel.
- Average throughput: Average rate of successfully broadcasted beacon messages across a communication channel.
- Overhead: The additional information exchanged between the vehicles to detect rogue nodes in the region.
- True positive rate: The percentage of rogue nodes is accurately detected and classified as rogue nodes.

$$\text{TPR} = \frac{\text{No. of rogue nodes detected correctly}}{\text{Total no. of rogue nodes}} \tag{3.11}$$

- False positive rate: The percentage of honest nodes is incorrectly detected and classified as rogue nodes.

$$\text{FPR} = \frac{\text{No. of honest nodes detected incorrectly}}{\text{Total no. of honest nodes}} \tag{3.12}$$

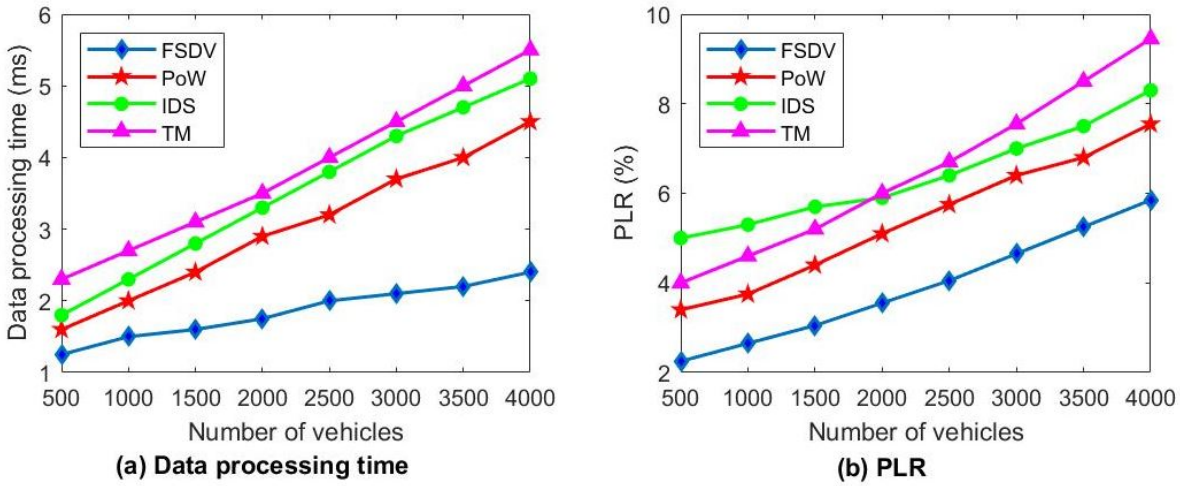


Figure 3.3: Comparison of urban scenarios of the FSDV framework with PoW, IDS, and TM schemes: (a) data processing time and (b) PLR.

3.6 Results

As mentioned in Section 3.4, we performed a simulation in two parts: the urban and highway scenarios. The vehicles in the urban scenario had a low mobility, while those in the highway scenario had a high mobility. During the simulation, the FSDV framework is analyzed based on the equations formulated in Section 3.3 and Section 3.4, and the results are presented below:

3.6.1 Urban Scenario

1) Data processing time: Data processing time is the time taken by the guard node to process and analyze the speed and position values received in beacon messages from the neighboring vehicles. The number of vehicles ranges from 500 to 4000 in an urban scenario. As the guard node adopts a delay-efficient fog computing technique to detect rogue and Sybil nodes in the region (Section 3.3), the data processing delay of our FSDV approach 43% is lower than PoW, IDS, and TM schemes. In the 4000–vehicle simulation, the data processing time was 42%, 53%, and 57% lower than that in the PoW, IDS, and TM schemes, respectively, as shown in Fig. 3.3a. The results show

that our FSDV framework is efficient, scalable, and can handle high vehicle densities.

2) PLR: From Fig 3.3b, it can be observed that PLR increases as the number of vehicles increases in the region. This is due to an increase in vehicles increasing the number of beacon messages received by the guard node, which in turn increases the load on the fog, resulting in packet drop or collision of some packets. However, the adoption of fog computing provides more bandwidth, resulting in 24% lower packet loss compared to [37,61,62] schemes. In the 4000–vehicle simulation, the PLR was 18%, 32%, and 38% lower than that in the PoW, IDS, and TEAM schemes, respectively.

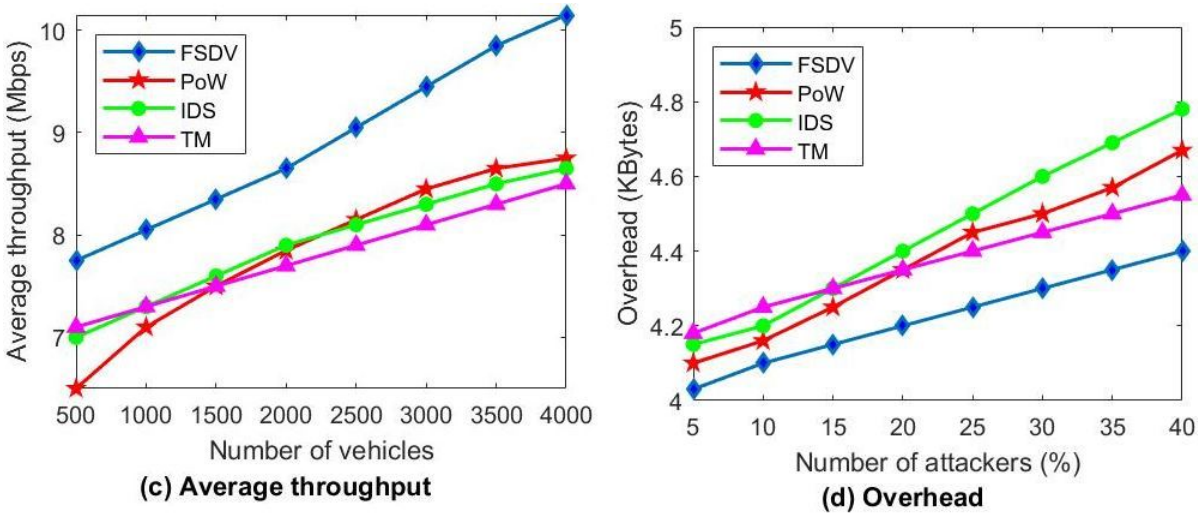


Figure 3.3: Comparison of urban scenarios of the FSDV framework with PoW, IDS, and TM schemes: (c) average throughput and (d) overhead.

3) Average throughput: The average throughput increases when the number of beacon messages is successfully broadcasted to all vehicles as the number of vehicles increases in the region, which measures the scalability of the proposed work. In FSDV, average throughput increases as vehicles increase from 500 to 4000 in an urban scenario. This was due to the scalability of our dynamic fog, low data processing delays (Fig. 3.3a), and low PLR (Fig. 3.3b) at all densities. In the 4000–vehicle simulation, the average throughput in an urban scenario was 16%, 19%, and 26%

higher than that in the PoW, IDS, and TM schemes, respectively, as shown in Fig. 3.3c.

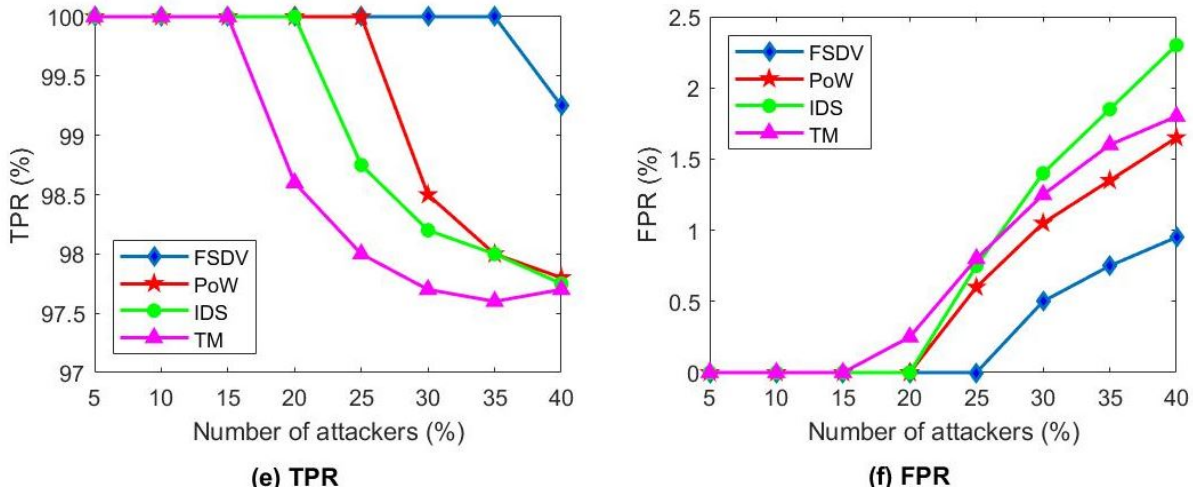


Figure 3.3: Comparison of urban scenarios of the FSDV framework with PoW, IDS, and TM schemes: (e) TPR and (f) FPR.

4) Overhead: Fig 3.3d shows the overhead of the FSDV framework ranging from 5% to 40% of rogue nodes in the region. In order to reduce overhead, instead of sending a challenge packet to all the vehicles in the region, the guard node sends it only when there is a suspicion of a Sybil attack resulting in 16% lower overhead compared to [37, 61, 62] schemes. For example, when the number of rogue nodes was 40% in the region, the overhead was 11%, 15%, and 9% lower than that of the PoW, IDS, and TM schemes, respectively.

5) TPR: TPR was calculated against the number of rogue nodes, as shown in Fig. 3.3e. When the number of rogue nodes was more than 35%, the TPR marginally decreased to 99.25%. Detecting the rogue node broadcasting malicious speed values was difficult when the speed variation was gradual. However, to generate a false congestion scenario and to simultaneously operate many active Sybil nodes within the region, the target node suddenly decreases the speed values. The FSDV performs better than PoW, IDS, and TM, up to 40% rogue nodes in the region. Thus, the FSDV framework can detect rogue nodes even at high vehicle densities, resulting in

a higher TPR compared to [37, 61, 62] schemes.

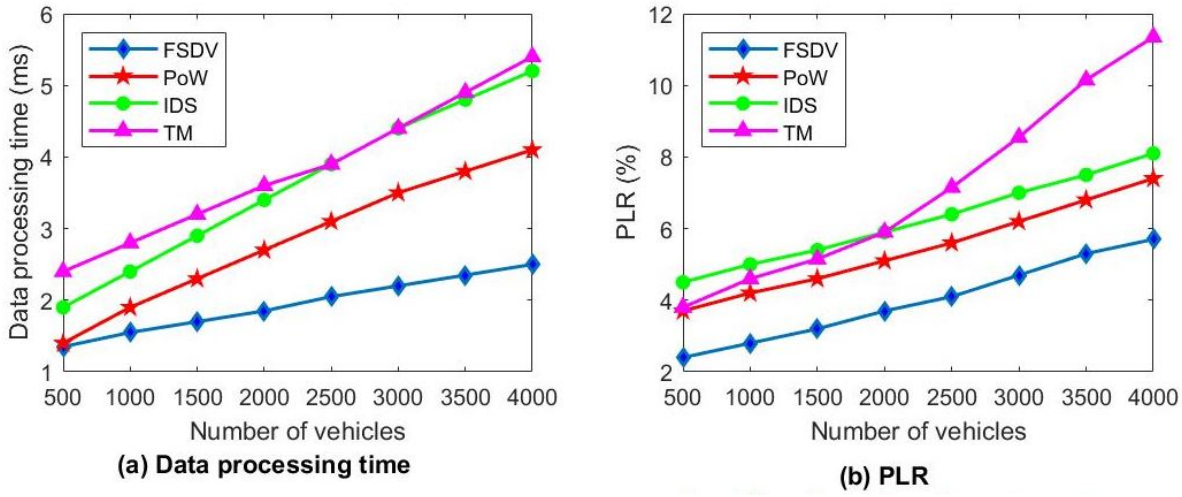


Figure 3.4: Comparison highway scenarios of the FSDV framework with PoW, IDS, and TM schemes: (a) data processing time and (b) PLR.

6) FPR: The increase in FPR deteriorates the performance of rogue node detection schemes by increasing the number of rogue nodes in the region. However, in the FSDV framework, in addition to comparing the difference in speed values received from beacon messages with dynamic threshold, we also perform the location-based beamforming technique to identify Sybil and rogue nodes resulting in 38% lower FPR even at 40% rogue nodes in the region compared to [37, 61, 62] schemes. For example, when the number of rogue nodes was 40% in the region, the FPR was 34%, 55%, and 41% lower than that of the PoW, IDS, and TM schemes, respectively as shown in Fig. 3.3f.

3.6.2 Highway Scenario

1) Data processing time: Fig. 3.4 a shows, when the number of vehicles increases from 500 to 4000, the data processing time increases as the guard node needs to process a large number of beacon messages received from all the vehicles in the region. However, as the OBUs of all vehicles are utilized in creating the dynamic fog layer, the

computation power of the guard node increases when the number of vehicles increases in the region results in a 40% lower processing delay compared to [37,61,62] schemes. In the 4000 vehicles simulation, the data processing time is 39%, 51%, and 54% lower than PoW, IDS, and TM schemes, respectively, as shown in Fig. 3.4a. The results show that our FSDV framework is efficient and can handle high vehicle densities.

2) PLR: PLR of the FSDV framework increases when the number of vehicles increases from 500 to 4000 due to the high mobility of the vehicles resulting in a collision of some packets. However, the PLR of our framework is lower compared to [37, 61, 62] at all vehicle densities. The PLR is calculated against the number of vehicles. In 4000 vehicles simulation, PLR is 22%, 29%, and 38% lower than the PoW, IDS, and TM schemes respectively, as shown in Fig. 3.4b.

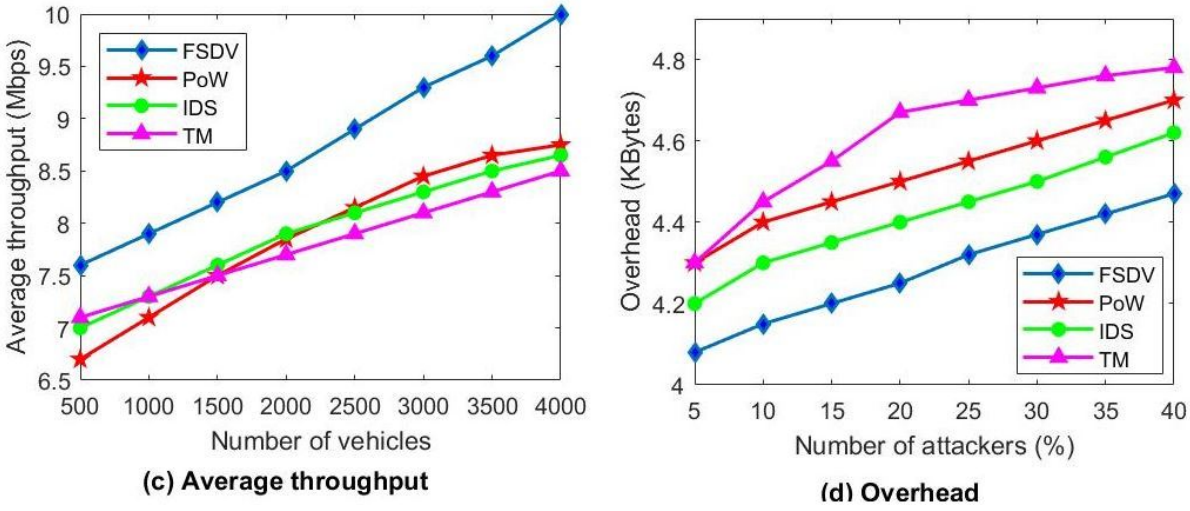


Figure 3.4: Comparison highway scenarios of the FSDV framework with PoW, IDS, and TM schemes: (c) average throughput and (d) overhead.

3) Average throughput: The average throughput is calculated against the number of vehicles and increases when the number of vehicles increases, as shown in Fig. 3.4c. Due to a large number of messages are successfully broadcasted to all vehicles region, the average throughput of the FSDV framework is higher compared to [37, 61, 62] schemes at all vehicle densities. In the 4000 vehicle simulation, average throughput

is 14%, 15%, and 19% higher than the PoW, IDS, and TM, respectively.

4) Overhead: The overhead of our framework is calculated against the number of rogue nodes (Fig. 3.4d). Finding the dynamic threshold increases the overhead of our framework. However, unlike existing approaches [37,61,62], our FSDV framework does not require any additional information, such as past vehicle data, trust score, and digital signature exchanged between the guard node, to detect rogue and Sybil nodes in the region resulting in 13% lower overhead in an urban scenario even when the number of rogue nodes increased up to 40% as shown in Fig. 3.3d. For a network with 40% rogue nodes, the overhead is 7%, 6%, and 10% lower than the PoW, IDS, and TM schemes, respectively.

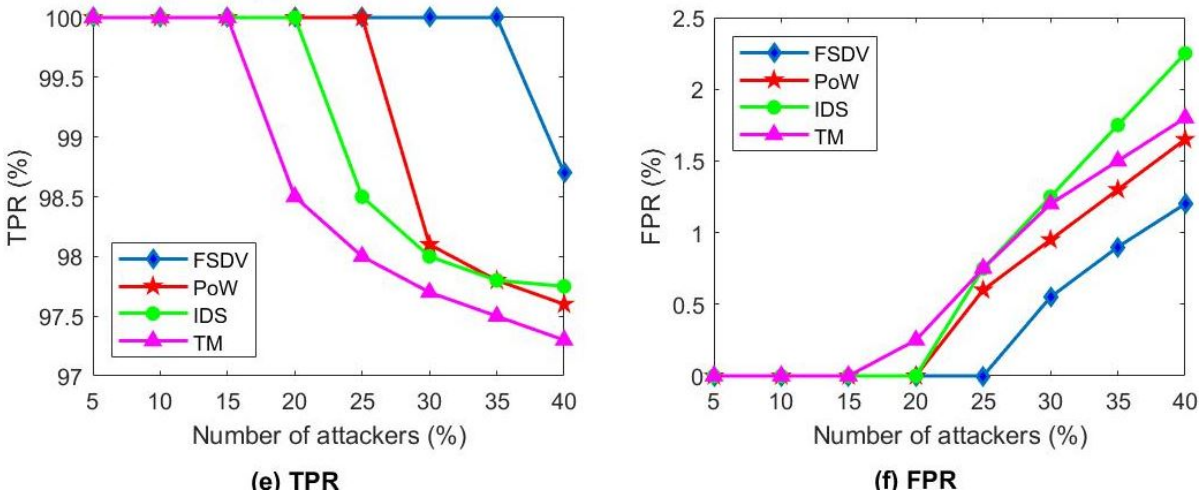


Figure 3.4: Comparison highway scenarios of the FSDV framework with PoW, IDS, and TM schemes: (e) TPR and (f) FPR.

5) TPR: Detecting the rogue nodes broadcasting false information is difficult if the speed varies gradually in the received beacon messages. However, to generate either fake traffic congestion or an accident, in addition to creating Sybil nodes, the target rogue node also decreases the speed values quickly. The FSDV algorithm also adapts location verification techniques to detect rogue nodes in the region resulting in detecting rogue nodes correctly (i.e.,100%) up to 30% rogue nodes in the region. It decreases slightly to 98.5% when the number of rogue nodes increases to 40%, as

shown in Fig. 3.4e. Moreover, the TPR of the FSDV framework is higher compared to PoW, IDS, and TM schemes at all vehicle densities.

6) FPR: The increase in FPR increases rogue nodes in the region as well as deteriorates the performance of the rogue node detection schemes. In the FSDV framework, the rogue nodes detection relies only on the speed and position values in beacon messages broadcasted by all vehicles in the region without taking into any trust scores or past vehicle data into consideration resulting in 35% lower FPR compared to [37, 61, 62] schemes. For a network with 40% rogue nodes, the FPR is 31%, 51%, and 38% lower than PoW, IDS, and TM schemes, respectively, as shown in Fig. 3.4f.

3.7 Summary

We studied the challenges in detecting Sybil attacks, such as high processing delay, high network overhead, high FPR, and low TPR, notably when the number of rogue nodes increases in the region at high vehicle densities. To address these limitations and to provide an efficient scheme to detect false messages broadcasted over the network, we proposed an OBU-based fog computing technique to detect rogue nodes increases by up to 40% in the region. The simulations were performed to evaluate the performance of the FSDV framework using OMNET++ and SUMO simulator. Results showed that the FSDV framework is scalable, efficient, robust, and performs up to 32% better than [37,61,62] techniques. Moreover, the FSDV framework ensures a 43% lower processing delay, 34% lower FPR, and 14% lower overhead at high vehicle densities compared to existing Sybil attack detection schemes [37, 61, 62]. Also, like F-RouND discussed in Chapter 2, FSDV framework does not depend on any roadside infrastructures like RSUs, or trust scores or past vehicle data in rogue nodes detection, which is a major advantage compared to existing schemes [37, 61, 62].

Chapter 4

Effect of Rogue nodes in Vehicular Platooning: Platoon Control Maneuver Attack

4.1 Introduction

The advancement in VANETs and cooperative adaptive cruise control (CACC) facilitates vehicles to organized into groups of close-following vehicles commonly known as a platoon. Platooning has been identified as a promising framework to improve road capacity, on-road safety, and energy efficiency through platoon management protocols [83, 84, 85]. A platoon is a complex physical system composed of a platoon leader and one or more followers, also known as platoon members. The platoon leader is usually the first vehicle of the platoon, which defines the speed, distance, acceleration, maximum deacceleration, and direction of platoon members; responsible for performing basic maneuvers like platoon formation, merging, splitting, etc., by exchanging various commands through beacon messages.

To maintain the stability of the platoon, the platoon leader broadcasts beacon messages to its members through one-vehicle look-ahead communication at a constant time interval (t) [84, 86]. In addition to that, each platoon member receives information such as speed, position, distance, and direction from its preceding vehicle. All the vehicles are equipped with radar to measure the distance and speed of the preceding vehicle. Studies and research on platooning have been in vogue for a number of years now. The most popular platooning research projects are California partners for advanced transit and highways (PATH), safe road trains for the environ-

ment (SARTRE), etc. Studies of these platooning research projects are focused with the aim of improving V2V communication for platooning, and CACC techniques [87].

Platoon in VANET highly depends on effective communication among platoon members to carry out basic platoon control maneuvers such as platoon merge and split. Failing to acquire communication and accomplish platoon merge and split maneuvers leads to the distortion of the platoon by the platoon leader. One such problem is accomplishing a platoon merge in the presence of strong interference caused by an unintended vehicle (i.e., rogue node) joining in the middle during the platoon merge maneuver. Michelle et al. and Yang et al. [88, 89] analyzed the interference caused by the rogue node during the platoon merge maneuver and determined whether the platoon merge maneuver can be performed or aborted based on scenarios like far truck interference, close truck interference, car interference, and channel impairments. The authors concentrated on providing a solution to accomplish the platoon merge maneuver in the presence of a rogue node in high vehicles dense regions like Manhattan and other downtown regions. Also, the proposed solution depends on the trust model and past vehicle data, which increases the overhead of the proposed techniques [88,89].

Amoozahdeh et al. [18] developed a platoon management protocol for VANET, which includes basic platooning maneuvers such as platoon merge and split. However, this approach has limitations such as high delay and high collision ratio during platoon merge. In addition, the proposed technique for the platoon merge performs only rear-end merge. Jeroen et al. [90] designed a hybrid controller for platoon merge and split maneuvers. The continuous-time system handles the longitudinal control, and a discrete-event supervisor decides on platoon merge and split maneuvers. However, this approach suffers from frequent loss of connection in high vehicle-dense regions, resulting in high PLR. Huang et al. [91] illustrated a cooperative platoon maneuver switching model for platoon merge and split operations based on hybrid automata and trust models. But, this protocol leads to high overhead and PLR.

The proposal of a novel intrusion detection framework is essential to detect rogue nodes and perform platoon merge in all possible scenarios in the VANET environment. Our *objective* is to accomplish platoon control maneuvers (i.e., platoon merge) even in the presence of strong interference caused by a rogue node joining the platoon during a merge maneuver. For example, consider Manhattan and other downtown regions, where vehicle density is higher than in the urban environment. Thus, the possibility of a rogue node entering the platoon gap is high, resulting in string instability that leads to aborting the platoon merge maneuver without being performed. In most extreme conditions, string instability cause vehicle collisions in the platoon.

The contribution of this work is two-fold. First, we have developed an algorithm called platoon merging for cooperative driving (PMCD) to identify an unintended vehicle entering the platoon gap and to accomplish platoon merge by splitting one large platoon into two or multiple sub-platoons in all possible scenarios, including high vehicle dense regions. Second, we have implemented our algorithm in VENTOS and SUMO simulators to measure the performance of PMCD. The novelty of PMCD lies in that our proposed technique dynamically adapts between interference and non-interference regions caused by rogue nodes in the platoon.

In this chapter, we have selected two different schemes for evaluation, Yang's approach [88], and Michelle's approach [89], and considered network throughput and PLR to measure the performance of PMCD with the help of ns-3 and SUMO simulators. To simulate the trace of vehicle movements, the SUMO simulator is used. The output of the SUMO simulator is given as input to the VENTOS simulator. VENTOS is a discrete event simulator that provides substantial support for the simulation of wired and wireless networks and also outputs trace files for every simulation. From the trace files, simulation data are collected and converted into graphs.

4.2 Related Works

This section presents an overview of the most recent existing schemes that describes platoon-based rogue node detection VANETs. Kremer et al. [92], and Hota et al. [93] proposed a trust-based scheme to detect rogue nodes in vehicular platoon merge scenarios. Based on the correctness of the data in beacon messages, positive or negative trust scores are assigned. Positive and negative trust values represent the normal and abnormal behavior of the vehicles in the platoon, respectively. When the calculated trust of any vehicle reaches a predefined threshold limit is known as a rogue node. Then, the information is broadcasted to all the vehicles in the region.

Zhou et al. [55] proposed a distributed collaborative intrusion detection framework that stores and compares past vehicle data to identify the false messages broadcasted by the rogue nodes in VANETs. The scheme proposed in [55] employs a clustering technique to segregate the vehicles based on the reputation state and behavior; then, the normal driving characteristics are compared with the individual clusters to detect malicious behaviors. The vehicles associated with the clusters exposing malicious behaviors, termed rogue nodes, are then ignored to contain the network damage. This approach [55] suffers from high processing delays in creating clusters and high PLR at high vehicle densities.

Pan et al. [94], and Xu et al. [95] proposed trust–based intrusion detection frameworks for rogue node detection. The models proposed in [94, 95] assign a trust value to all the vehicles in the platoon region based on their behavior. Vehicles that drop or alter the message are considered rogue nodes with negative trust values. Vehicles other than the rogue nodes are honest vehicles with positive trust values. The trust values of the vehicles are maintained in the score table, which gets updated whenever messages are broadcasted by the vehicles. Each vehicle has a copy of the score table containing information about all the vehicles in the region. Thus, the messages

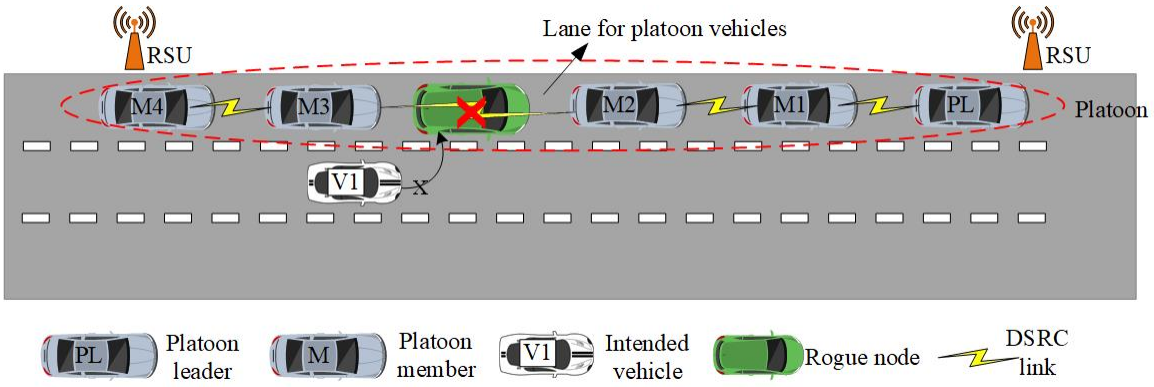


Figure 4.1: Interference caused by an unintended vehicle entered into a platoon in dense vehicle regions.

received from the rogue nodes are ignored to contain network damage. As such, the frameworks proposed in [94, 95] suffer from high PLR and high overhead in detecting the rogue nodes.

To overcome the limitations of the existing schemes [88,89], we propose the PMCD framework, which is used to identify the presence of a rogue node in the platoon based on the PLR and to accomplish the platoon merge in all possible scenarios.

4.3 Proposed PMCD Framework

This section illustrates the working principle of the PMCD framework.

4.3.1 Problem Description

The effect of the interference problem in the platoon merge maneuver is discussed in this section. In Fig. 4.1, PL is the platoon leader, and M1, M2, M3, and M4 are the platoon members. Vehicle look-ahead communication is used to exchange information, such as speed, distance, direction, etc., among the platoon members through beacon messages. RSUs are used to specify the lane assigned for platooned vehicles and to inform each PL of the optimal size of a platoon based on various characteristics like road length, vehicle density, etc. [18] Assume PL receives a merge request from

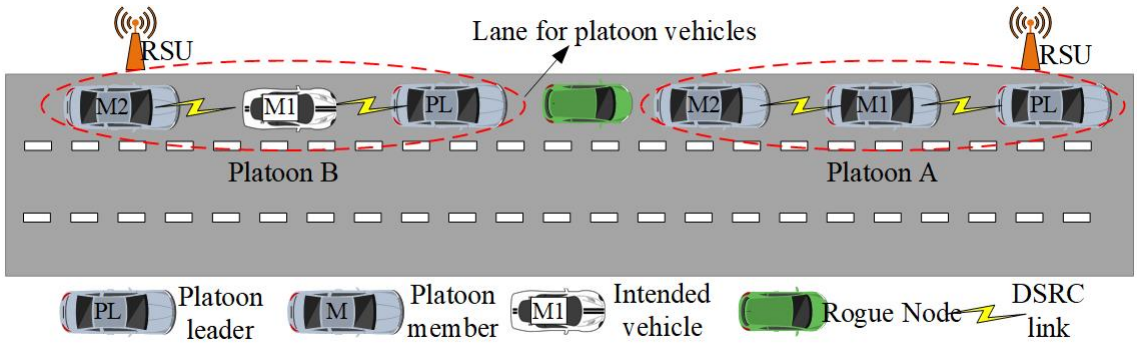


Figure 4.2: Execution of platoon merge maneuver in the presence of the rogue node by splitting the platoon into sub platoons i.e. platoon A and platoon B.

the intended vehicle V1; upon receiving the merge request, the PL checks the location of V1 and sends a slow-down command to M3 through a multihop technique. M3 creates a gap to let V1 join the platoon. However, a rogue node might change the lane and enter the platoon, leading to communication distortion among M2 and M3 as the rogue node does not participate in platoon communication, resulting in high PLR during the platoon merge maneuver execution. This situation causes instability in the platoon and also forces the PL to abort the platoon merge maneuver in the middle without being performed. Finding a solution for the interference problem plays an essential role in accomplishing platoon merge in all possible scenarios, including high vehicle dense regions. Subsection 4.3.2 provide a solution to the interference problem in the platoon environment.

4.3.2 Rogue Node Detection Technique

Consider the scenario discussed in Section 4.3.1, where the rogue node entering the middle of the platoon disrupts communication among platoon members, resulting in instability of the platoon and aborting the platoon merge maneuver. PMCD provides an efficient solution to solve this problem. The first step in PMCD is to identify whether the vehicle entered into the platoon gap is the intended vehicle or not. This can be done by comparing the radar distance with a GPS position in beacon messages.

Table 4.1: Notations used in PMCD algorithm.

Variables	Purpose
P_j	Platoon(s) in a given scenario
PL	Platoon leader of the platoon P_j
$PM_{1...n}$	Platoon members of the platoon P_j
$V1$	Vehicle sends merge request to join the platoon P_j
Th_{PLR}	Acceptable PLR

If a rogue node enters the platoon, the radar distance will not be coherent with the GPS position broadcast by the car in front as the rogue vehicle does not take part in platoon communication. Moreover, a rogue node in the middle of a platoon leads to communication distortion among the platoon members, resulting in high PLR during the platoon merge maneuver execution [88, 89].

The second step of PMCD is to provide a solution to this problem by splitting the platoon at the point where a rogue node entered into two or more sub-platoons. Fig. 4.2 depicts a solution to the scenario discussed in Section 4.3.1 by splitting a platoon into two sub platoons: platoon A and platoon B. PL, M1, and M2 form a new platoon: platoon A. M3 is considered as a PL for platoon B, the intended vehicle joins the platoon B and become the first member of the platoon B (M1), and M4 is considered as the second member of the platoon B (M2). DSRC communication is used to exchange various messages among platoon members to carry out the split operation. RSU informs the optimal size of platoon A and platoon B to appropriate PLs.

4.3.3 PMCD Algorithm

PMCD algorithm illustrates the step-by-step process of execution of the platoon merge maneuver. In the PMCD algorithm, the participating candidates are PL, PM, and $V1$. The platoon merge is initiated when a PL receives a merge request from $V1$ (line 1). Upon receiving a request, PL executes a function select_PM() to

Algorithm 3: PMCD: Rogue node identification and platoon merge algorithm

Input: P_j, $V1$, PL, $PM_{1...n}$

Output: Extended P or sub platoons P_j

```
1  if (PL receives merge_request from V1) then
2      Repeat
3      PL ⇐ select_PM_i() ;
4      PM_i ⇐ platoon_merge() ;
5      if (gap_open() == True) then
6          lane change(V1) ;
7          PM_i ⇐ compare (radar_distance, GPS_position) ;
8      end
9      if (radar_distance == GPS_position) then
10         reduce_gap(PM_i) ;
11         update_info(PL) ;
12     end
13     else
14         calculate_PLR() ;
15     end
16     if (PLR > Th_PLR && PM > 2) then
17         platoon_split() ;
18         P_j ⇐ platoon_formation(PM_1, PM_{i-1}) ;
19         P_{j+1} ⇐ platoon_formation(PM_i, PM_n) ;
20         if (PLR > Th_PLR && PM == 2) then
21             platoon_split() ;
22             lane change(V1) ;
23             platoon_merge() ;
24         end
25     end
26     Until radar_distance == GPS_position
27 end
```

check the location of $V1$ and thereby select the suitable PM to carry out the platoon merge operation (line 3). The selected PM (PM_i) performs the platoon merge operation by invoking the platoon_merge() function (line 4). PM_i creates a gap to let $V1$ join the platoon (lines 5 and 6). Once the vehicle enters the platoon, PM_i compares the GPS position and radar distance to verify whether the $V1$ joined the platoon or not (line 9).

If the GPS position and radar distance match, then it is clear that $V1$ entered the platoon, and thus, PM_i reduces a gap to complete the merge process (lines 9 and 10). On completing the platoon merge maneuver successfully, PL receives a message from PMs through a multihop technique (line 11). If a rogue node enters the platoon, the radar distance will not be coherent with the GPS position broadcast by the car in front, as the rogue node does not participate in platoon communication. PLR is calculated to confirm the rogue node enters the platoon gap (line 14). PLR higher than the minimum threshold indicates an unintended vehicle joins the platoon, and thus, platoon_split() function is invoked to split the large platoon into two sub-platoons (lines 16 to 19). The algorithm is repeated until the platoon merge maneuver is accomplished. PMCD also handles a special case, the presence of a rogue node in the two-vehicle platoon. First, the platoon_split function is invoked to split the vehicles in a platoon into individual vehicles. After the successful split of the two-vehicle platoon, $V1$ performs the platoon merge maneuver with any of the split vehicles and forms a new platoon (lines 20 to 23). The purpose of the variables used in the PMCD algorithm is represented in Table 4.1.

4.4 Mathematical Model Analysis

The impact of the rogue node entering the platoon leads to instability, oscillation, and deviation (Section 4.3). In this model, we measure the position deviation of the

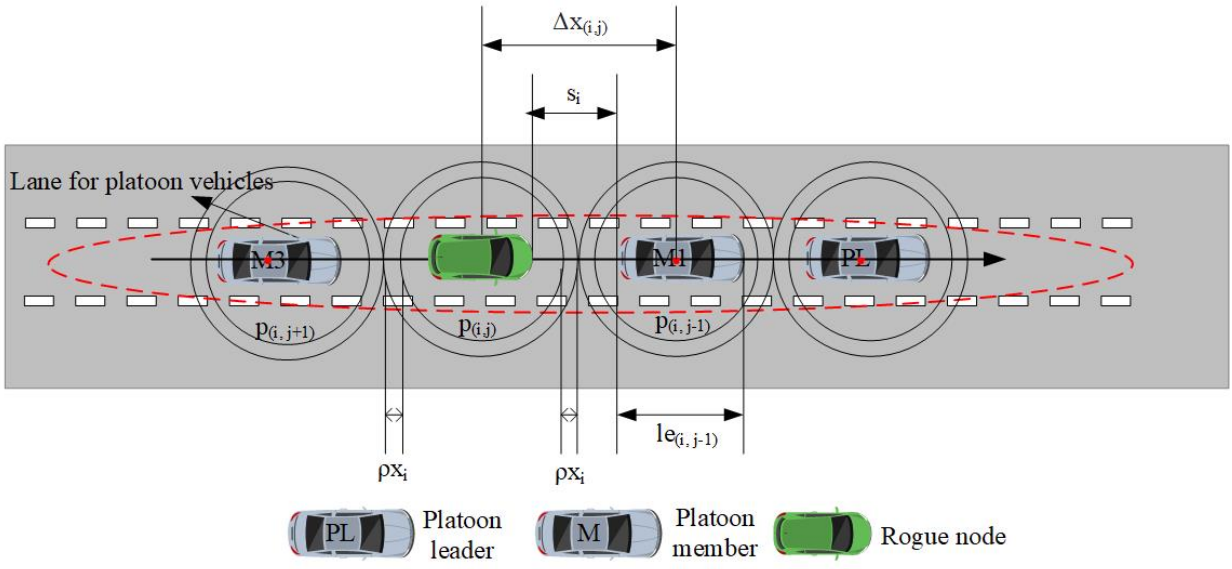


Figure 4.3: An example scenario of the PMCD mathematical modeling to detect the rogue node in the platoon.

platoon members in the presence of a rogue node.

The platoon's information flow determines the neighboring vehicles' position as input to the vehicle dynamics model. $\Delta x_{(i,j)}(t)$ as position differences between two vehicles $p_{(i,j)}$ and $p_{(i,j-1)}$ from their center of gravity, depicted in Fig. 4.3. The $\Delta x_{(i,j)}(t)$ can be defined as:

$$\Delta x_{(i,j)}(t) = x_{(i,j)}(t) - x_{(i,j-1)}(t) \tag{4.1}$$

The dynamic platoon model for platoon stability incorporates that spacing and velocity errors between vehicles. Based on (4.1), the platoon stability goal can mathematically be expressed as:

$$\Delta x_{(i,j)}(t) - \frac{1}{2(le_{(i,j)} + le_{(i,j-1)})} = s_i \tag{4.2}$$

$$v_{(i,j)}(t) = v_{(i,0)}(t),$$
$$\forall j \in [1, N]$$

Where $le_{(i,j-1)}$ and $le_{(i,j)}$ denote the length of vehicles $v_{(i,j-1)}$ and $v_{(i,j)}$, respectively. Based on (4.2), the error dynamics model is given by:

$$\Delta x_{(i,j)}(t) - \frac{1}{2(le_{(i,j)} + le_{(i,j-1)})} - s_i = pe \tag{4.3}$$

$$v_{(i,j)}(t) - v_{(i,0)}(t) = se$$

Where $pe_{(i,j)}$ represents the position error, which is the difference between relative position, desired space, and length between vehicle $p_{(i,j)}$ and $p_{(i,j-1)}$ in the platoon i. The velocity error is $se_{(i,j)}$. The $x_{(i,j)}$ must not exceed the interval limit that would be:

$$\phi x_i = \phi y_i = \frac{s_i}{4} \tag{4.4}$$

Position error is considered severe in the platoon as it could lead to a frequent loss of communication resulting in either vehicle collision or break up of the platoon.

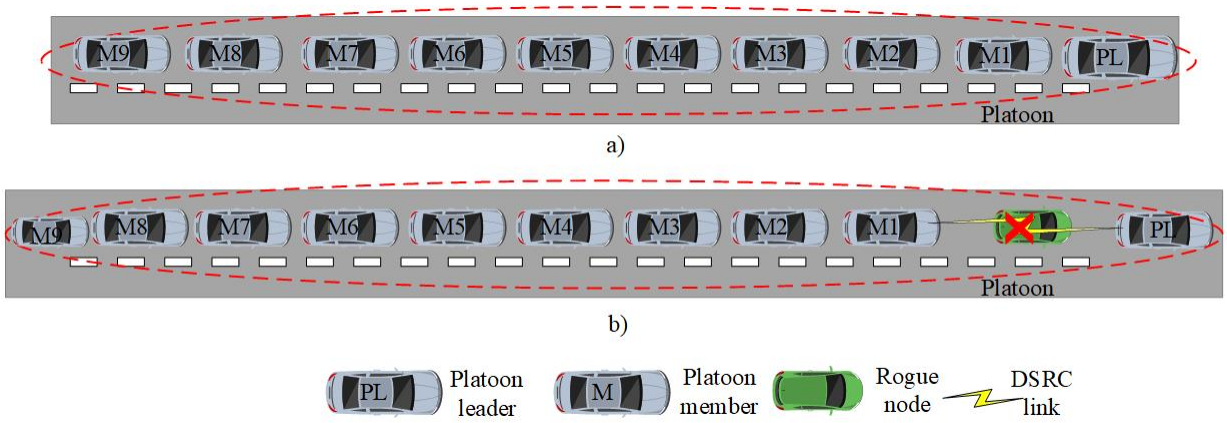


Figure 4.4: A sample scenario of ten-vehicle platoon on a highway scenario: a) vehicles follow each other smoothly, and b) a rogue node enters the platoon during the platoon merge maneuver.

4.5 Performance Evaluation

This section evaluates and analyzes the performance of the PMCD framework discussed in Section 4.3 and Section 4.4.

4.5.1 Simulation Setup

The simulations are carried out using VENTOS and SUMO simulators. SUMO is an open-source traffic-events simulator that provides a trace of vehicle movements at the end of every simulation for a map imported from OpenStreetMap. VENTOS is a discrete event simulator used to measure the performance of the network using the platoon deployment model, platoon mobility model, etc., [96, 97, 98]. To perform the simulation, we imported the highways of Norman, Oklahoma, into the SUMO simulator. The output of the SUMO simulator is given as input to the VENTOS simulator to accomplish platoon merge in the presence of a rogue node. To assess the scalability and behavior of the PMCD framework, we increase the platoon size up to 10 and the presence of a rogue node in the network.

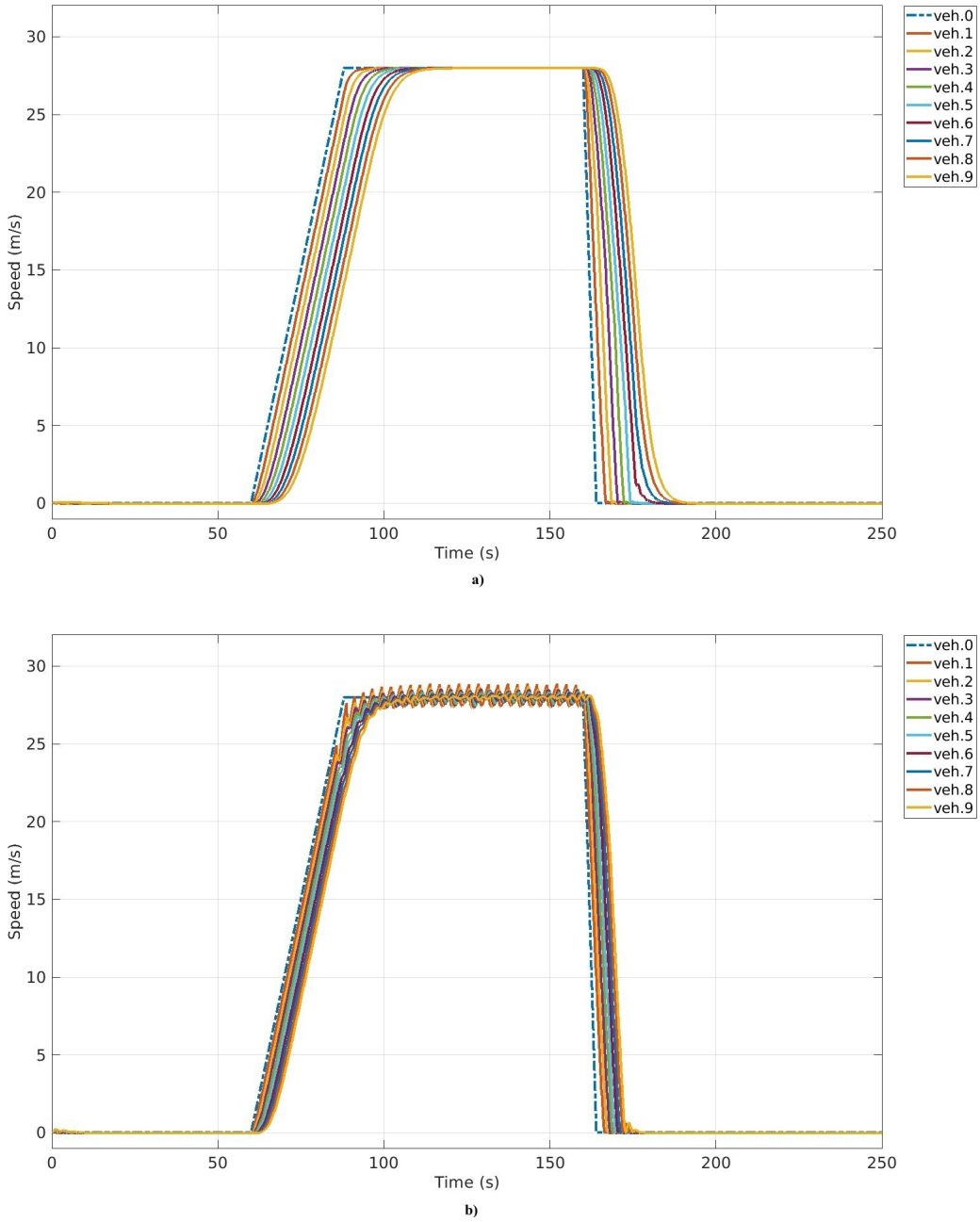


Figure 4.5: Speed profile of vehicle stream with ten-vehicle platoon: a) vehicles speeds up and slow down, all vehicles follow each other properly and b) string stability is not maintained in the presence of a rogue node leads to distortion.

Table 4.2: Parameters used in Simulation of the PMCD Framework

Parameters	**Values**
Platoon Size	2-10
Road length	5 Miles
Number of lanes	2
Vehicle speed	30-65 Miles/hr
Transmission range	500 m
Beacon message size	256 bytes
Protocol	IEEE802.11p

4.5.2 Impact of the Rogue Node in the Platoon

We simulated the scenario of a rogue node in the platoon discussed in Section 4.2. Fig. 4.5a shows the speed profile of a ten-vehicle platoon moving on a straight single-lane highway when no rogue node is present, as shown in Fig 4.4a. The simulation starts at t=60 s; the vehicle accelerates to 28 m/s and smoothly decelerates to 0 m/s. Fig 4.5b presents the speed profile of the same vehicle stream in the presence of a rogue node causing strong interference by entering the platoon during the platoon merge maneuver, as shown in Fig. 4.5a.

The platoon leader broadcasts beacon messages to its members to maintain the stability of the platoon. However, the rogue node enters the platoon at t=80 s, leading to communication distortion, resulting in high PLR and instability in the platoon, as depicted in Fig. 4.5b. This rogue node behavior could lead to catastrophic consequences like vehicle collision. To overcome the effect of the rogue node, the platoon dismantled, and all the vehicles simply followed the car-following model to reach the destination. As a result, we have developed PMCD, which provides an efficient solution to the interference problem and accomplishes platoon merge in all possible scenarios (Section 4.4).

4.5.3 Performance Metrics

The simulations were performed based on the algorithm discussed in Section 4.3. Table 4.2 summarizes the most commonly used parameters used in the simulation. We considered the following metrics to evaluate the performance of our PMCD framework and to compare our results with Yang and Michelle's schemes:

- Network Throughput: The rate of successfully broadcasted beacon messages across a communication channel.
- PLR: The ratio of the number of lost packets to the total number of packets sent across a communication channel.
- Collision Ratio: The number of packets colliding across a network before reaching the destination
- Overhead: The additional information exchanged between the vehicles to detect rogue node in the region.

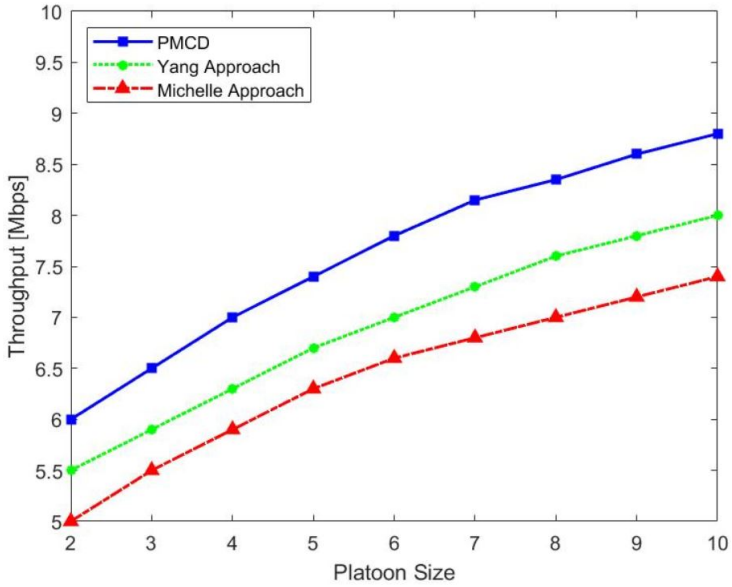
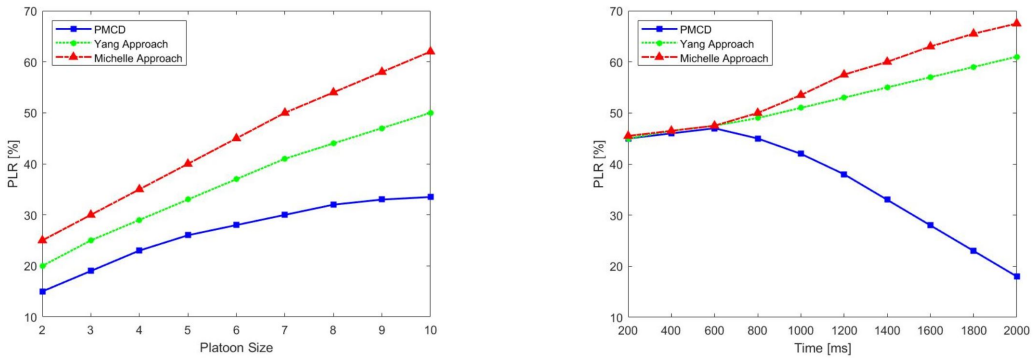


Figure 4.6: Comparison of network throughput with Yang's approach and Michelle's approach by varying platoon size.



(a) (b)

Figure 4.7: Comparison of PLR with Yang's approach and Michelle's approach: (a) varying platoon size, (b) varying time [ms].

4.6 Results

We considered network throughput and PLR to measure the performance of the PMCD algorithm (Section 4.3) and to compare our results with Yang's approach [89] and Michelle's approach [88]. The simulations were performed using VENTOS and SUMO simulators. The results are as follows:

1) Network Throughput: Network throughput is the number of packets successfully transmitted across a network at a given time interval (t). It is calculated against the number of vehicles in a platoon, represented in Fig. 4.6. The larger the number of vehicles in a platoon, the larger the number of messages that needs to be transmitted across a network to accomplish the platoon merge maneuver. As the PMCD algorithm discussed in Section 4.3 provides an efficient solution to platoon merge maneuvers in all possible scenarios, including the presence of a rogue node in the middle of the platoon, the number of successfully transmitted messages increases as the number of vehicles increases in the platoon. Thus, PMCD provides high network throughput at all vehicle densities.

2)PLR: PLR is the ratio of the number of lost packets to the total number of packets sent in a platoon. In Figure 4.7a, PLR is calculated against platoon size, i.e., the number of vehicles in a platoon. PLR of PMCD was observed to be lower at all vehicle densities due to the successful execution of the platoon merge maneuver, even in the presence of a rogue node in the middle of a platoon. It increases marginally as the number of vehicles increases due to the collision of some packets. However, the PLR of PMCD is overall 47% lower compared to existing approaches (i.e., Yang's approach [89] and Michelle's approach [88]) at all vehicle densities. To observe the time taken by PMCD to accomplish platoon split in the presence of a rogue node in a platoon, we calculated PLR against simulation time, represented in Figure 4.7b. PLR of PMCD was observed to be high in the beginning, as it takes 200 to 400

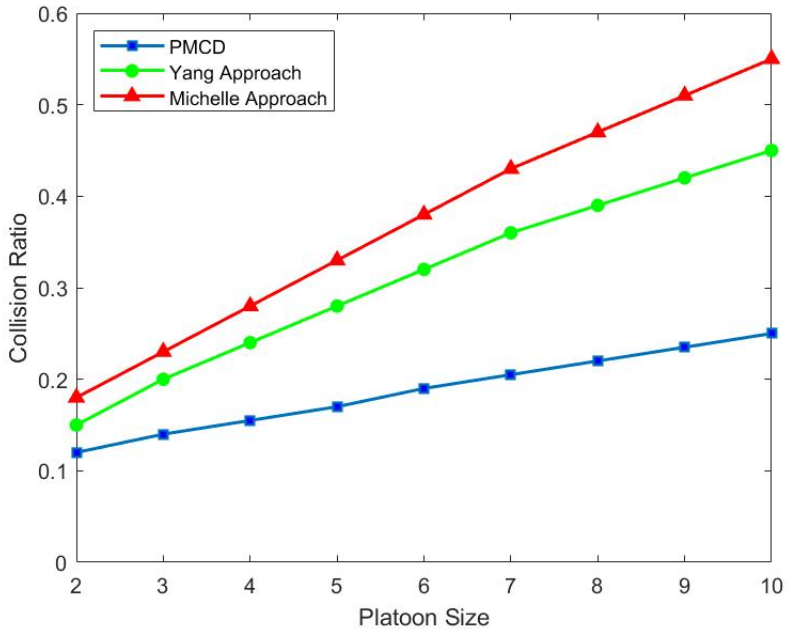


Figure 4.8: Comparison of collision ratio with Yang's approach and Michelle's approach by varying platoon size.

ms to detect interference in a platoon. However, the PMCD algorithm discussed in Section 4.3 provides an efficient solution to accomplish platoon merge by splitting a large platoon into two or more sub-platoons. Thus, the PLR of PMCD decreases gradually, and it observed be to lower with an increase in simulation time.

3) Collision Ratio We performed this experiment at a time interval *(t)* to observe the number of packets colliding before reaching the destination. We observed that the collision ratio of PMCD is lower at high vehicle densities. It increases slightly as the number of platoon sizes increases in the system, as shown in Fig. 4.8. It is due to the additional packets generated being more likely to encounter another packet, resulting in a collision. When the platoon size was 10, the collision ratio was 20% and 26% lower than that in Yang's and Michelle's schemes, respectively [88, 89].

4) Overhead: The overhead of the PMCD framework was calculated against the platoon size and increases as the platoon sizes increase in the system due to the time taken to detect whether the rogue nodes were correctly identified. However, unlike

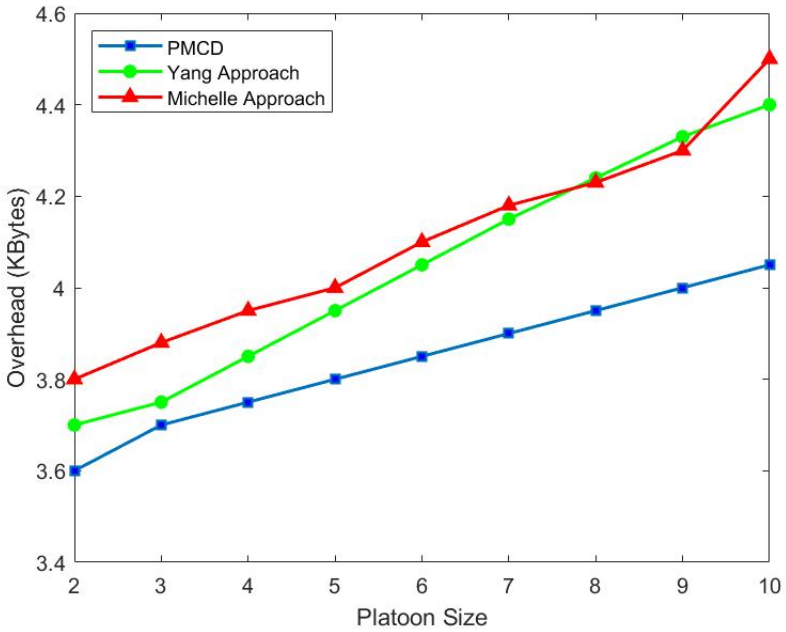


Figure 4.9: Comparison of overhead with Yang's approach and Michelle's approach by varying platoon size.

Yang's and Michelle's approaches [88,89], our PMCD framework detects the presence of a rogue node at all possible scenarios resulting in 20% lower overhead compared to Yang's and Michelle's approaches at high vehicle densities, as shown in Fig 4.9.

4.7 Summary

In this chapter, we studied the challenges of the presence of the rogue node in the platoon. To address the challenges and to provide an efficient to accomplish platoon control maneuvers, such as merging and splitting in the platoon, we have developed an algorithm called PMCD. PMCD identifies the strong interference caused by the rogue node entering the platoon gap and accomplishes platoon merge by splitting one large platoon into two or more sub-platoons.

Moreover, PMCD ensures platoon merge in all possible scenarios, including highly dense vehicle regions like downtown regions, where the possibility of a rogue node

entering the platoon is high. We have analyzed the performance of PMCD through an effective simulation using VENTOS and SUMO simulators. PMCD ensures 47% lower PLR and 20% lower overhead compared to [88, 89] techniques. The results showed that PMCD is robust, efficient, and accomplishes platoon merge compared to Yang's and Michelle's approaches at all vehicle densities.

Chapter 5

Conclusions and Future Directions

In this research, we studied some of the most vulnerable security attacks in VANETs: False information attack, Sybil attack, and platoon control maneuvers attack, and improved the performance of the rogue nodes detection algorithms by addressing the limitations of the previous works. The researchers have proposed different mechanisms to detect rogue nodes using trust score, reputation, and cryptographic techniques. However, difficulties arise for highly mobile and dynamic networks like VANETs due to their inherent nature. Therefore, we have provided statistical and traffic flow theory-based models that utilize only beacon messages to detect rogue nodes. The key contributions of this dissertation are summarized as follows:

1. **False Information Attack:** F-RouND has been proposed in Chapter 2, which utilizes traffic flow theory and hypothesis testing to detect rogue nodes in VANETs responsible for the false information attack using the fog computing technique. The performance of the F-RouND has been compared with existing approaches through an extensive simulation in both highway and urban scenarios. Results show that the F-RouND framework demonstrated the effectiveness of the fog computing technique in determining rogue nodes, even when the number of rogue nodes increases by up to 40% in the region. Moreover, the performance of our extensive hypothesis test in validating the rogue nodes ensured 45% lower processing delay, 36% lower FPR, and 12% lower overhead at the urban scenario and 44% lower processing delay, 32% lower FPR, and 10% lower overhead at the highway scenario compared to the existing techniques [36,37,38].

2. **Sybil Attack:** A novel FSDV framework has been proposed in Chapter 3 to detect rogue nodes responsible for Sybil attacks based on dynamic threshold and location information broadcasted in beacon messages using the fog computing technique. FSDV has the ability to detect the rogue nodes and their generated Sybil nodes. Extensive simulations were performed based on the highway and urban scenarios to evaluate the FSDV framework's performance and compare the results with the existing approaches [37, 61, 62]. Results showed that the FSDV framework is scalable, efficient, robust, and performs up to 32% better than [37, 61, 62] techniques. Moreover, the FSDV framework ensures a 43% lower processing delay, 34% lower FPR, and 14% lower overhead at high vehicle densities compared to existing Sybil attack detection schemes.

3. **Platoon Control Maneuver Attack:** Similar to previous rogue node detection techniques discussed in Chapters 2 and 3, an intrusion detection technique (PMCD) has been proposed to detect rogue nodes responsible for causing strong interference in joining the platoon during the merging maneuver. PMCD ensures platoon merge in all possible scenarios, including highly dense vehicle regions like downtown regions, where the possibility of a rogue node entering the platoon is high. The performance of the PMCD algorithm and its analytical expressions have been validated through an extensive simulation. The results showed that PMCD is scalable, reduces overall PLR by 47%, and accomplishes platoon merge compared to existing approaches [88, 89], even at high vehicle densities.

The proposed rogue node detection frameworks and related analysis will help cybersecurity and software engineers working on cooperative driving build efficient schemes to detect security attacks and improve quality of service (QoS), such as delay, throughput, and overhead. Also, the proposed frameworks could be further extended

in a number of ways. Some of them are listed below:

1. **Cross layer-based Intrusion Detection Scheme for Enhancing Security in VANETs using Fog Computing:** Cross layer design provides stability and scalability to create robust and efficient communication protocols. The main objective of any cross-layer scheme is to leverage the information between layers, thus enhancing the performance of attack detection frameworks [99, 100]. Developing a fog-based intrusion detection system for VANETs using a cross-layer detection technique helps to detect rogue nodes involved in various security attacks: message distortion, GPS spoofing, timing, black hole attack, and wormhole attacks and also improves the QoS of our proposed frameworks.

2. **Next generation of VANETs using Fog Computing for 6G and Beyond Real Time Applications:** By leveraging 6G and beyond enabled VANETs, flexible communication can be achieved among vehicles with ultra-high reliability and low latency. Applications in 6G and beyond VANETs rely on sharing mobile data among vehicles, which is still challenging due to the enormous data volume and the prohibitive cost of transmitting such data using 6G networks [101, 102]. Therefore, it is imperative to investigate and design a framework for efficient cooperative data sharing in fog computing-assisted 6G and beyond VANETs. Fog computing techniques provide a significant amount of computing, storage, and resources near 6G and beyond VANETs, which could be used as a solution to the problems of capacity and latency, supplying vehicles with computing and storage resources.

3. **Novel Deep Learning methods for Internet of Things (IoT):** IoT is a vision for an internetwork of intelligent, communicating objects, which is on the cusp of transforming human lives. The convergence of technologies like ubiquitous wireless communications, deep learning, real-time analytics, and em-

bedded systems has made novel IoT applications possible in a multitude of domains [103]. Deep learning techniques such as convolution neural networks and long-short-term memory have been widely used in developing security for IoT applications. Advanced driver assistance applications in autonomous cars are heavily based on deep learning techniques that perform forward-collision warning, blind-spot detection, traffic sign recognition, and traffic safety [104, 105]. However, enhancing the performance and efficiency of these deep learning techniques is one of the big challenges for implementing secure real-time applications. Thus, developing efficient and secure deep-learning models could enhance the performance of existing IoT applications.

The most direct extension of this work is to extend the proposed methods according to coexistence of different heterogeneous networks and analyze the effects of not only coexistence of the networks but also their heterogeneity levels on the performance. All in all, this research helps the readers along the illuminating journey to understand the implications of security attacks in VANETs through various rogue node detection techniques.

Bibliography

[1] Saif Al-Sultan, Moath M Al-Doori, Ali H Al-Bayatti, and Hussien Zedan. A comprehensive survey on vehicular ad hoc network. *Journal of network and computer applications*, 37:380–392, 2014.

[2] Azlan Awang, Khaleel Husain, Nidal Kamel, and Sonia Aissa. Routing in vehicular ad-hoc networks: A survey on single-and cross-layer design techniques, and perspectives. *IEEE Access*, 5:9497–9517, 2017.

[3] Hannes Hartenstein and LP Laberteaux. A tutorial survey on vehicular ad hoc networks. *IEEE Communications magazine*, 46(6):164–171, 2008.

[4] Manisha Chahal, Sandeep Harit, Krishn K Mishra, Arun Kumar Sangaiah, and Zhigao Zheng. A survey on software-defined networking in vehicular ad hoc networks: Challenges, applications and use cases. *Sustainable cities and society*, 35:830–840, 2017.

[5] Vinita Jindal and Punam Bedi. Vehicular ad-hoc networks: introduction, standards, routing protocols and challenges. *International Journal of Computer Science Issues (IJCSI)*, 13(2):44, 2016.

[6] Sherali Zeadally, Ray Hunt, Yuh-Shyan Chen, Angela Irwin, and Aamir Hassan. Vehicular ad hoc networks (vanets): status, results, and challenges. *Telecommunication Systems*, 50:217–241, 2012.

[7] Hakima Khelifi, Senlin Luo, Boubakr Nour, Hassine Moungla, Yasir Faheem, Rasheed Hussain, and Adlen Ksentini. Named data networking in vehicular ad

hoc networks: State-of-the-art and challenges. *IEEE Communications Surveys & Tutorials*, 22(1):320–351, 2019.

[8] Keyvan Golestan, Ayman Jundi, L Nassar, Farook Sattar, Fakhri Karray, M Kamel, and Slim Boumaiza. Vehicular ad-hoc networks (vanets): capabilities, challenges in information gathering and data fusion. In *Autonomous and Intelligent Systems: Third International Conference, AIS 2012, Aveiro, Portugal, June 25-27, 2012. Proceedings*, pages 34–41. Springer, 2012.

[9] Reza Ghebleh. A comparative classification of information dissemination approaches in vehicular ad hoc networks from distinctive viewpoints: A survey. *Computer Networks*, 131:15–37, 2018.

[10] Wai Chen, Ratul K Guha, Taek Jin Kwon, John Lee, and Yuan-Ying Hsu. A survey and challenges in routing and data dissemination in vehicular ad hoc networks. *Wireless Communications and Mobile Computing*, 11(7):787–795, 2011.

[11] Hasan Ali Khattak, Saif Ul Islam, Ikram Ud Din, and Mohsen Guizani. Integrating fog computing with vanets: A consumer perspective. *IEEE Communications Standards Magazine*, 3(1):19–25, 2019.

[12] Ammara Anjum Khan, Mehran Abolhasan, and Wei Ni. 5g next generation vanets using sdn and fog computing framework. In *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–6. IEEE, 2018.

[13] Anirudh Paranjothi, Mohammad S Khan, and Mohammed Atiquzzaman. Dfcv: a novel approach for message dissemination in connected vehicles using dynamic fog. In *Wired/Wireless Internet Communications: 16th IFIP WG 6.2*

International Conference, WWIC 2018, Boston, MA, USA, June 18–20, 2018, Proceedings, pages 311–322. Springer, 2018.

[14] Ata Ullah, Shumayla Yaqoob, Muhammad Imran, and Huansheng Ning. Emergency message dissemination schemes based on congestion avoidance in vanet and vehicular fog computing. *IEEE Access*, 7:1570–1585, 2018.

[15] Md Julkar Nayeen Mahi, Sudipto Chaki, Shamim Ahmed, Milon Biswas, Shamim Kaiser, Mohammad Shahidul Islam, Mehdi Sookhak, Alistair Barros, and Md Whaiduzzaman. A review on vanet research: Perspective of recent emerging technologies. *IEEE Access*, 2022.

[16] Sean Joe Taylor, Farhan Ahmad, Hoang Nga Nguyen, and Siraj Ahmed Shaikh. Vehicular platoon communication: architecture, security threats and open challenges. *Sensors*, 23(1):134, 2023.

[17] Shunyuan Xiao, Xiaohua Ge, Qing-Long Han, and Yijun Zhang. Resource-efficient platooning control of connected automated vehicles over vanets. *IEEE Transactions on Intelligent Vehicles*, 7(3):579–589, 2022.

[18] Mani Amoozadeh, Hui Deng, Chen-Nee Chuah, H Michael Zhang, and Dipak Ghosal. Platoon management with cooperative adaptive cruise control enabled by vanet. *Vehicular communications*, 2(2):110–123, 2015.

[19] Mushtaq Ahmad, Zahid Khan, Anis Koubaa, and Wadii Boulila. A microscopic platoon stability model using vehicle-to-vehicle communication. *Electronics*, 11(13):1994, 2022.

[20] Eman Mousavinejad and Ljubo Vlacic. Secure platooning control of automated vehicles under cyber attacks. *ISA transactions*, 127:229–238, 2022.

[21] Qianwen Li, Zhiwei Chen, and Xiaopeng Li. A review of connected and automated vehicle platoon merging and splitting operations. *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[22] Muawia Abdelmagid Elsadig and Yahia A Fadlalla. Vanets security issues and challenges: A survey. *Indian Journal of Science and Technology*, 9(28):1–8, 2016.

[23] Sunilkumar S Manvi and Shrikant Tangade. A survey on authentication schemes in vanets for secured communication. *Vehicular Communications*, 9:19–30, 2017.

[24] David Antolino Rivas, José M Barceló-Ordinas, Manel Guerrero Zapata, and Julián D Morillo-Pozo. Security on vanets: Privacy, misbehaving nodes, false information and secure data aggregation. *Journal of Network and Computer Applications*, 34(6):1942–1955, 2011.

[25] Avleen Kaur Malhi, Shalini Batra, and Husanbir Singh Pannu. Security of vehicular ad-hoc networks: A comprehensive survey. *Computers & Security*, 89:101664, 2020.

[26] Muhammad Sameer Sheikh, Jun Liang, and Wensong Wang. A survey of security services, attacks, and applications for vehicular ad hoc networks (vanets). *Sensors*, 19(16):3589, 2019.

[27] Maria Azees, Pandi Vijayakumar, and Lazarus Jegatha Deborah. Comprehensive survey on security services in vehicular ad-hoc networks. *IET Intelligent Transport Systems*, 10(6):379–388, 2016.

[28] Anirudh Paranjothi, Mohammad S Khan, Rizwan Patan, Reza M Parizi, and Mohammed Atiquzzaman. Vanetomo: A congestion identification and control

scheme in connected vehicles using network tomography. *Computer Communications*, 151:275–289, 2020.

[29] Mohammad Naderi, Farzad Zargari, and Mohammad Ghanbari. Adaptive beacon broadcast in opportunistic routing for vanets. *Ad Hoc Networks*, 86:119–130, 2019.

[30] Pranav Kumar Singh, Anup Agarwal, Gaurav Nakum, Danda B Rawat, and Sukumar Nandi. Mpfslp: Masqueraded probabilistic flooding for source-location privacy in vanets. *IEEE Transactions on Vehicular Technology*, 69(10):11383–11393, 2020.

[31] Xiaozhen Lu, Xiaoyue Wan, Liang Xiao, Yuliang Tang, and Weihua Zhuang. Learning-based rogue edge detection in vanets with ambient radio signals. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2018.

[32] George Loukas, Eirini Karapistoli, Emmanouil Panaousis, Panagiotis Sarigiannidis, Anatolij Bezemskij, and Tuan Vuong. A taxonomy and survey of cyber-physical intrusion detection approaches for vehicles. *Ad Hoc Networks*, 84:124–147, 2019.

[33] Richard Gilles Engoulou, Martine Bellaïche, Samuel Pierre, and Alejandro Quintero. Vanet security surveys. *Computer Communications*, 44:1–13, 2014.

[34] Sunil M Sangve, Reena Bhati, and Vidhya N Gavali. Intrusion detection system for detecting rogue nodes in vehicular ad-hoc network. In *2017 International Conference on Data Management, Analytics and Innovation (ICDMAI)*, pages 127–131. IEEE, 2017.

[35] Sparsh Sharma and Ajay Kaul. A survey on intrusion detection systems and honeypot based proactive security mechanisms in vanets and vanet cloud. *Vehicular communications*, 12:138–164, 2018.

[36] Basmah Al-Otaibi, Najla Al-Nabhan, and Yuan Tian. Privacy-preserving vehicular rogue node detection scheme for fog computing. *Sensors*, 19(4):965, 2019.

[37] Kamran Zaidi, Milos B Milojevic, Veselin Rakocevic, Arumugam Nallanathan, and Muttukrishnan Rajarajan. Host-based intrusion detection for vanets: A statistical approach to rogue node detection. *IEEE transactions on vehicular technology*, 65(8):6703–6714, 2015.

[38] Farhan Ahmad, Virginia NL Franqueira, and Asma Adnane. Team: A trust evaluation and management framework in context-enabled vehicular ad-hoc networks. *IEEE Access*, 6:28643–28660, 2018.

[39] Anirudh Paranjothi, Urcun Tanik, Yuehua Wang, and Mohammad S Khan. Hybrid-vehfog: a robust approach for reliable dissemination of critical messages in connected vehicles. *Transactions on Emerging Telecommunications Technologies*, 30(6):e3595, 2019.

[40] Lina Bariah, Dina Shehada, Ehab Salahat, and Chan Yeob Yeun. Recent advances in vanet security: a survey. In *2015 IEEE 82nd vehicular technology conference (VTC2015-fall)*, pages 1–7. IEEE, 2015.

[41] Sahil Garg, Amritpal Singh, Kuljeet Kaur, Gagangeet Singh Aujla, Shalini Batra, Neeraj Kumar, and Mohammad S Obaidat. Edge computing-based security framework for big data analytics in vanets. *IEEE Network*, 33(2):72–81, 2019.

[42] Anirudh Paranjothi, Mohammad S Khan, and Mohammed Atiquzzaman. Hybrid-vehcloud: An obstacle shadowing approach for vanets in urban environment. In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pages 1–5. IEEE, 2018.

[43] Muhammad Arshad, Zahid Ullah, Muhammad Khalid, Naveed Ahmad, Waqar Khalid, Duri Shahwar, and Yue Cao. Beacon trust management system and fake data detection in vehicular ad-hoc networks. *IET Intelligent Transport Systems*, 13(5):780–788, 2019.

[44] Junwei Liang, Jianyong Chen, Yingying Zhu, and Richard Yu. A novel intrusion detection system for vehicular ad hoc networks (vanets) based on differences of traffic flow and position. *Applied Soft Computing*, 75:712–727, 2019.

[45] Hichem Sedjelmaci, Sidi Mohammed Senouci, and Mosa Ali Abu-Rgheff. An efficient and lightweight intrusion detection mechanism for service-oriented vehicular networks. *IEEE Internet of things journal*, 1(6):570–577, 2014.

[46] Saneeha Ahmed, Sarab Al-Rubeaai, and Kemal Tepe. Novel trust framework for vehicular networks. *IEEE Transactions on Vehicular Technology*, 66(10):9498–9511, 2017.

[47] Chunhua Zhang, Kangqiang Chen, Xin Zeng, and Xiaoping Xue. Misbehavior detection based on support vector machine and dempster-shafer theory of evidence in vanets. *IEEE Access*, 6:59860–59870, 2018.

[48] Erfan A Shams, Ahmet Rizaner, and Ali Hakan Ulusoy. Trust aware support vector machine intrusion detection and prevention system in vehicular ad hoc networks. *Computers & Security*, 78:245–254, 2018.

[49] Pravin Mundhe, Vijay Kumar Yadav, Shekhar Verma, and S Venkatesan. Efficient lattice-based ring signature for message authentication in vanets. *IEEE Systems Journal*, 14(4):5463–5474, 2020.

[50] Li Yang, Abdallah Moubayed, Ismail Hamieh, and Abdallah Shami. Tree-based intelligent intrusion detection system in internet of vehicles. In *2019 IEEE global communications conference (GLOBECOM)*, pages 1–6. IEEE, 2019.

[51] Kuldeep Narayan Tripathi and Subhash Chander Sharma. A trust based model (tbm) to detect rogue nodes in vehicular ad-hoc networks (vanets). *International Journal of System Assurance Engineering and Management*, 11:426–440, 2020.

[52] Tarak Nandy, Rafidah Md Noor, Mohd Yamani Idna Bin Idris, and Sananda Bhattacharyya. T-bcids: Trust-based collaborative intrusion detection system for vanet. In *2020 National Conference on Emerging Trends on Sustainable Technology and Engineering Applications (NCETSTEA)*, pages 1–5. IEEE, 2020.

[53] Jizhao Liu, Heng Pan, Junbao Zhang, Qian Zhang, and Qiusheng Zheng. Detecting bogus messages in vehicular ad-hoc networks: an information fusion approach. In *Wireless Sensor Networks: 11th China Wireless Sensor Network Conference, CWSN 2017, Tianjin, China, October 12-14, 2017, Revised Selected Papers 11*, pages 191–200. Springer, 2018.

[54] Praveensankar Manimaran et al. Ndnids: An intrusion detection system for ndn based vanet. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pages 1–5. IEEE, 2020.

[55] Man Zhou, Lansheng Han, Hongwei Lu, and Cai Fu. Distributed collaborative intrusion detection system for vehicular ad hoc networks based on invariant. *Computer Networks*, 172:107174, 2020.

[56] Ayoob Ayoob, Ghaith Khalil, Morshed Chowdhury, and Robin Doss. Intrusion detection system classifier for vanet based on pre-processing feature extraction. In *Future Network Systems and Security: 5th International Conference, FNSS 2019, Melbourne, VIC, Australia, November 27–29, 2019, Proceedings 5*, pages 3–22. Springer, 2019.

[57] Wilmer Arellano and Imad Mahgoub. Trafficmodeler extensions: A case for rapid vanet simulation using, omnet++, sumo, and veins. In *2013 High Capacity Optical Networks and Emerging/Enabling Technologies*, pages 109–115. IEEE, 2013.

[58] Christoph Sommer, Reinhard German, and Falko Dressler. Bidirectionally coupled network and road traffic simulation for improved ivc analysis. *IEEE Transactions on mobile computing*, 10(1):3–15, 2010.

[59] Zaid A Abdulkader, Azizol Abdullah, Mohd Taufik Abdullah, and Zuriati Ahmad Zukarnain. A survey on sybil attack detection in vehicular ad hoc networks (vanet). *Journal of Computers*, 29(2):1–6, 2018.

[60] Dhia Eddine Laouiti, Marwane Ayaida, Nadhir Messai, Sameh Najeh, Leila Najjar, and Ferdaous Chaabane. Sybil attack detection in vanets using an adaboost classifier. In *2022 International Wireless Communications and Mobile Computing (IWCMC)*, pages 217–222. IEEE, 2022.

[61] Mohamed Baza, Mahmoud Nabil, Mohamed MEA Mahmoud, Niclas Bewermeier, Kemal Fidan, Waleed Alasmary, and Mohamed Abdallah. Detecting sybil attacks using proofs of work and location in vanets. *IEEE Transactions on Dependable and Secure Computing*, 19(1):39–53, 2020.

[62] Marwane Ayaida, Nadhir Messai, Sameh Najeh, and Kouamé Boris Ndjore. A macroscopic traffic model-based approach for sybil attack detection in vanets. *Ad Hoc Networks*, 90:101845, 2019.

[63] Bo Yu, Cheng-Zhong Xu, and Bin Xiao. Detecting sybil attacks in vanets. *Journal of Parallel and Distributed Computing*, 73(6):746–756, 2013.

[64] Xia Feng, Chun-yan Li, De-xin Chen, and Jin Tang. A method for defensing against multi-source sybil attacks in vanet. *Peer-to-Peer Networking and Applications*, 10:305–314, 2017.

[65] Celestine Iwendi, Mueen Uddin, James A Ansere, Pascal Nkurunziza, Joseph Henry Anajemba, and Ali Kashif Bashir. On detection of sybil attack in large-scale vanets using spider-monkey technique. *IEEE Access*, 6:47258–47267, 2018.

[66] Sarra Benadla and Omar Rafik Merad-Boudia. The impact of sybil attacks on vehicular fog networks. In *2021 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI)*, pages 1–6. IEEE, 2021.

[67] Federico Concone, Fabrizio De Vita, Ajay Pratap, Dario Bruneo, Giuseppe Lo Re, and Sajal K Das. A fog-assisted system to defend against sybils in vehicular crowdsourcing. *Pervasive and Mobile Computing*, 83:101612, 2022.

[68] Carlos HOO Quevedo, Ana MBC Quevedo, Gustavo A Campos, Rafael L Gomes, Joaquim Celestino, and Ahmed Serhrouchni. An intelligent mechanism for sybil attacks detection in vanets. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2020.

[69] Khaled Rabieh, Mohamed MEA Mahmoud, Terry N Guo, and Mohamed Younis. Cross-layer scheme for detecting large-scale colluding sybil attack in vanets.

In *2015 IEEE International Conference on Communications (ICC)*, pages 7298–7303. IEEE, 2015.

[70] Binod Kumar Pattanayak, Omkar Pattnaik, and Sasmita Pani. Dealing with sybil attack in vanet. In *Intelligent and Cloud Computing: Proceedings of ICICC 2019, Volume 1*, pages 471–480. Springer, 2021.

[71] Mandeep Kaur Saggi and Ranjeet Kaur. Isolation of sybil attack in vanet using neighboring information. In *2015 IEEE International Advance Computing Conference (IACC)*, pages 46–51. IEEE, 2015.

[72] Ye Chen, Yingxu Lai, Zhaoyi Zhang, Hanmei Li, and Yuhang Wang. Mdfd: A multi-source data fusion detection framework for sybil attack detection in vanets. *Computer Networks*, page 109608, 2023.

[73] Saleh Khalaj Monfared and Saeed Shokrollahi. Darvan: A fully decentralized anonymous and reliable routing for vanets. *Computer Networks*, page 109561, 2023.

[74] Santosh Kumar, Amol Vasudeva, and Manu Sood. Sybil attack countermeasures in vehicular ad hoc networks. In *2022 International Conference on Communications, Information, Electronic and Energy Systems (CIEES)*, pages 1–6. IEEE, 2022.

[75] Samira Tahajomi Banafshehvaragh and Amir Masoud Rahmani. Intrusion, anomaly, and attack detection in smart vehicles. *Microprocessors and Microsystems*, 96:104726, 2023.

[76] Mohammed Lamine Bouchouia, Houda Labiod, Ons Jelassi, Jean-Philippe Monteuuis, Wafa Ben Jaballah, Jonathan Petit, and Zonghua Zhang. A survey

on misbehavior detection for connected and autonomous vehicles. *Vehicular Communications*, page 100586, 2023.

[77] Xiang Liu, Weiwei Wu, Wanyuan Wang, Yuhang Xu, Xiumin Wang, and Helei Cui. Budget-feasible sybil-proof mechanisms for crowdsensing. In *Frontiers of Algorithmic Wisdom: International Joint Conference, IJTCS-FAW 2022, Hong Kong, China, August 15–19, 2022, Revised Selected Papers*, pages 269–288. Springer, 2023.

[78] Sofia Azam, Maryum Bibi, Rabia Riaz, Sanam Shahla Rizvi, and Se Jin Kwon. Collaborative learning based sybil attack detection in vehicular ad-hoc networks (vanets). *Sensors*, 22(18):6934, 2022.

[79] Seyed Salar Sefati and Sara Ghiasi Tabrizi. Detecting sybil attack in vehicular ad-hoc networks (vanets) by using fitness function, signal strength index and throughput. *Wireless Personal Communications*, pages 1–21, 2022.

[80] Ammar Haydari and Yasin Yilmaz. Rsu-based online intrusion detection and mitigation for vanet. *Sensors*, 22(19):7612, 2022.

[81] Tejaswi Sapala, Rama Chandra Suresh Reddy Penumallu, Reddy Sai Kiran, MV Rajesh, and BS Kiruthika Devi. A survey on vanet attacks and its security mechanisms. In *2022 Seventh International Conference on Parallel, Distributed and Grid Computing (PDGC)*, pages 435–440. IEEE, 2022.

[82] Sarath Babu and Arun Raj Kumar P. A comprehensive survey on simulators, emulators, and testbeds for vanets. *International Journal of Communication Systems*, 35(8):e5123, 2022.

[83] Hao Hu, Rongxing Lu, Zonghua Zhang, and Jun Shao. Replace: A reliable trust-based platoon service recommendation scheme in vanet. *IEEE Transactions on Vehicular Technology*, 66(2):1786–1797, 2016.

[84] Elis Kulla, Ningling Jiang, Evjola Spaho, and Noritaka Nishihara. A survey on platooning techniques in vanets. In *Complex, Intelligent, and Software Intensive Systems: Proceedings of the 12th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS-2018)*, pages 650–659. Springer, 2019.

[85] Caixing Shao, Supeng Leng, Yan Zhang, Alexey Vinel, and Magnus Jonsson. Performance analysis of connectivity probability and connectivity-aware mac protocol design for platoon-based vanets. *IEEE Transactions on Vehicular Technology*, 64(12):5596–5609, 2015.

[86] Yang Zhang and Guohong Cao. V-pada: Vehicle-platoon-aware data access in vanets. *IEEE Transactions on Vehicular Technology*, 60(5):2326–2339, 2011.

[87] Philipp Kremer, Ipsita Koley, Soumyajit Dey, and Sangyoung Park. State estimation for attack detection in vehicle platoon using vanet and controller model. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8. IEEE, 2020.

[88] Michele Segata, Bastian Bloessl, Stefan Joerer, Falko Dressler, and Renato Lo Cigno. Supporting platooning maneuvers through ivc: An initial protocol analysis for the join maneuver. In *2014 11th Annual conference on wireless on-demand network systems and services (WONS)*, pages 130–137. IEEE, 2014.

[89] Shiyan Yang, Steven E Shladover, Xiao-Yun Lu, John Spring, David Nelson, and Hani Ramezani. A first investigation of truck drivers on-the-road experience using cooperative adaptive cruise control. 2018.

[90] Jeroen Ploeg, Elham Semsar-Kazerooni, Alejandro I Morales Medina, Jan FCM de Jongh, Jacco van de Sluis, Alexey Voronov, Cristofer Englund, Reinder J Bril, Hrishikesh Salunkhe, Álvaro Arrúe, et al. Cooperative automated maneuvering at the 2016 grand cooperative driving challenge. *IEEE Transactions on Intelligent Transportation Systems*, 19(4):1213–1226, 2017.

[91] Zichao Huang, Duanfeng Chu, Chaozhong Wu, and Yi He. Path planning and cooperative control for automated vehicle platoon using hybrid automata. *IEEE Transactions on Intelligent Transportation Systems*, 20(3):959–974, 2018.

[92] Daniel Kyalo Ndambuki and Hitmi Khalifa Alhitmi. Attack mitigation and security for vehicle platoon. *Journal of Cyber Security and Mobility*, pages 497–530, 2022.

[93] Lopamudra Hota, Biraja Prasad Nayak, Bibhudatta Sahoo, Peter HJ Chong, and Arun Kumar. An adaptive traffic-flow management system with a cooperative transitional maneuver for vehicular platoons. *Sensors*, 23(5):2481, 2023.

[94] Chengwei Pan, Yong Chen, Songge Chen, and Ikram Ali. Event-based distributed fixed-time resilient control for heterogeneous vehicular platoon against attack and disturbances. *IEEE Internet of Things Journal*, 2023.

[95] Yuanyuan Xu, Kun Zhu, Hu Xu, and Jiequ Ji. Deep reinforcement learning for multi-objective resource allocation in multi-platoon cooperative vehicular networks. *IEEE Transactions on Wireless Communications*, 2023.

[96] Thales Teixeira de Almeida, Lucas de Carvalho Gomes, Fernando Molano Ortiz, José Geraldo Ribeiro Júnior, and Luís Henrique MK Costa. Comparative analysis of a vehicular safety application in ns-3 and veins. *IEEE Transactions on Intelligent Transportation Systems*, 23(1):620–629, 2020.

[97] Jesús Mena-Oreja and Javier Gozalvez. Permit-a sumo simulator for platooning maneuvers in mixed traffic scenarios. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3445–3450. IEEE, 2018.

[98] Fuchun Liu, Yun He, and Junhong Zhao. Analysis of fuel economy and pollutant emissions of autonomous vehicle platoon based on plexe. In *2018 Chinese Automation Congress (CAC)*, pages 640–645. IEEE, 2018.

[99] Fadlallah Chbib, Sherali Zeadally, Rida Khatoun, Lyes Khoukhi, Walid Fahs, and Jamal Haydar. A secure cross-layer architecture for reactive routing in vehicle to vehicle (v2v) communications. *Vehicular Communications*, 38:100541, 2022.

[100] Raghad Baiad, Omar Alhussein, Hadi Otrok, and Sami Muhaidat. Novel cross layer detection schemes to detect blackhole attack against qos-olsr protocol in vanet. *Vehicular Communications*, 5:9–17, 2016.

[101] Pandi Vijayakumar, Maria Azees, Sergei A Kozlov, and Joel JPC Rodrigues. An anonymous batch authentication and key exchange protocols for 6g enabled vanets. *IEEE Transactions on Intelligent Transportation Systems*, 23(2):1630–1638, 2021.

[102] K Suresh Kumar, AS Radha Mani, S Sundaresan, and T Ananth Kumar. Modeling of vanet for future generation transportation system through edge/fog/cloud computing powered by 6g. *Cloud and IoT-based vehicular ad hoc networks*, pages 105–124, 2021.

[103] Yuhao Wang, Vlado Menkovski, Ivan Wang-Hei Ho, and Mykola Pechenizkiy. Vanet meets deep learning: The effect of packet loss on the object detection performance. In *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, pages 1–5. IEEE, 2019.

[104] Zhe Peng, Shang Gao, Zecheng Li, Bin Xiao, and Yi Qian. Vehicle safety improvement through deep learning and mobile sensing. *IEEE network*, 32(4):28–33, 2018.

[105] Shrikant Tangade, Sunilkumar S Manvi, and Stive Hassan. A deep learning based driver classification and trust computation in vanets. In *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, pages 1–6. IEEE, 2019.

Author's List of Publications

[1] Anirudh Paranjothi, Mohammad S. Khan, Mais Nijim, and Rajab Challoo. MAvanet: Message authentication in VANET using social networks. *In 2016 IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 1-8, IEEE, 2016.

[2] Anirudh Paranjothi, Mohammad S. Khan, and Mais Nijim. Survey on three components of mobile cloud computing: offloading, distribution and privacy. *Journal of Computer and Communications*, 5:1-15, 2017.

[3] Anirudh Paranjothi, Mohammad S. Khan, and Mohammed Atiquzzaman. DFCV: a novel approach for message dissemination in connected vehicles using dynamic fog. *In WiredWireless Internet Communications: 16th IFIP WG 6.2 International Conference, WWIC 2018, Boston, MA, USA, June 18-20, 2018, Proceedings*, pages 311-322, Springer International Publishing, 2018.

[4] Anirudh Paranjothi, Mohammad S. Khan, and Mohammed Atiquzzaman. Hybrid-Vehcloud: An obstacle shadowing approach for vanets in urban environment. *In 2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pages 1-5, IEEE, 2018.

[5] Yasmin Jahir, Mohammed Atiquzzaman, Hazem Refai, Anirudh Paranjothi, and Peter G. LoPresti. Routing protocols and architecture for disaster area network: A survey. *Ad Hoc Networks*, 82:1-14, 2019.

[6] Anirudh Paranjothi, Urcun Tanik, Yuehua Wang, and Mohammad S. Khan. Hybrid Vehfog: a robust approach for reliable dissemination of critical messages in

connected vehicles. *Transactions on Emerging Telecommunications Technologies*, 30(6):1-15, 2019.

[7] Anirudh Paranjothi, Mohammad S. Khan, Sherali Zeadally, Ajinkya Pawar, and David Hicks. GSTR: Secure multihop message dissemination in connected vehicles using social trust model. *Internet of Things*, 7:1-18, 2019.

[8] Anirudh Paranjothi, Mohammed Atiquzzaman, and Mohammad S. Khan. PMCD: Platoon Merging approach for cooperative driving. *Internet Technology Letters*, 3(1): 1-5, 2020.

[9] Anirudh Paranjothi, Mohammad S. Khan, Rizwan Patan, Reza M. Parizi, and Mohammed Atiquzzaman. VANETomo: A congestion identification and control scheme in connected vehicles using network tomography. *Computer Communications*, 151:275-289, 2020.

[10] Anirudh Paranjothi, Mohammad S. Khan, and Sherali Zeadally. A survey on congestion detection and control in connected vehicles. *Ad Hoc Networks*, 108:102-120, 2020.

[11] Anirudh Paranjothi, Mohammed Atiquzzaman, and Mohammad S. Khan. F-RouND: Fog based Rogue Nodes Detection in Vehicular Ad hoc Networks. *In GLOBECOM 2020 IEEE Global Communications Conference*, pages 1-6, IEEE, 2020.

[12] Anirudh Paranjothi, Mohammed Atiquzzaman, and Mohammad S. Khan. Message Dissemination in Connected Vehicles. *In Connected and Autonomous Vehicles in Smart Cities,* edited by Hussein T. Mouftah, Melike Erol-Kantarci, Sameh Sorour, CRC Press, pages 203-222, 2020.

[13] Anirudh Paranjothi,and Mohammed Atiquzzaman. Enhancing security in vanets with efficient sybil attack detection using fog computing. *In ICC 2021 IEEE International Conference on Communications*, pages 1-6, IEEE, 2021.

[14] Anirudh Paranjothi, and Mohammed Atiquzzaman. A statistical approach for enhancing security in VANETs with efficient rogue node detection using fog computing. *Digital Communications and Networks*, 8(5):814-824, 2021.