Real-time Defense Strategies against Rogue Nodes in Vehicular Ad Hoc Networks

A Dissertation
Presented in Partial Fulfillment of the Requirements for the
Degree of Doctor of Philosophy
with a
Major in Computer Science
in the
College of Graduate Studies
University of Idaho
by
Mohamed Sobhy Mahmoud Mohamed

Major Professor: Axel Krings, Ph.D.
Committee Members:
Robert Rinker, Ph.D.;
Fredrick Sheldon, Ph.D.;
Ahmed Abdel-Rahim, Ph.D.
Department Administrator: Fredrick Sheldon, Ph.D.

August 2018

Authorization to Submit Dissertation

This dissertation of Mohamed Sobhy Mahmoud Mohamed, submitted for the degree of Doctor of Philosophy with a Major in Computer Science and titled "Real-time Defense Strategies against Rogue Nodes in Vehicular Ad Hoc Networks" has been reviewed in final form. Permission, as indicated by the signatures and dates below, is now granted to submit final copies to the College of Graduate Studies for approval.

Major Professor:	Axel Krings, Ph.D.	Date
Committee Members:	Robert Rinker, Ph.D.	Date
	Fredrick Sheldon, Ph.D	Date
	Ahmed Abdel-Rahim, Ph.D.	Date
Department Administrator:	Fredrick Sheldon, Ph.D	Date

Abstract

Intelligent Transportation Systems provide technologies, services, and applications that allow wireless communication between vehicles, and between vehicles and the roadside infrastructure using Dedicated Short Range Communications (DSRC). This collaborative communication forms a Vehicular Ad-Hoc Network (VANET). The most important applications in VANET are DSRC Safety Applications, which play a significant role in reducing road accidents. As these applications rely on wireless communication they inherit all of the associated security problems. VANET security is crucial, since application reliability, and thus safety, must not be compromised. Rogue nodes can severely affects the reliability of applications by sharing or injecting false data in the network. They may even launch Sybil attacks, in which a rogue node pretends to be multiple nodes by impersonating their identities, potentially causing serious problems.

In this dissertation, defense strategies against rogue nodes in vehicular networks are proposed, which improve the reliability of safety applications operating in hostile environments. These strategies take advantage of certain properties of a new hybrid jammer, which is introduced to cause safety applications to fail. First, a detection algorithm for the new hybrid jammer is presented. Then an enhanced voting-based algorithm is shown, which is capable of significantly reducing the decision times of safety applications to alert drivers. Next, passive and active Sybil attack detection algorithms are introduced, which enhance the resilience against Sybil attacks in static and dynamic power environments respectively. Finally, an enhanced active Sybil detection algorithm is introduced and its impact on DSRC safety applications is analyzed.

The defense strategies and their effectiveness were investigated using simulations and field experiments. The latter used vehicles equipped with commercial DSRC equipment. The experimental results show that the defense strategies have significant impact on enhancing the reliability of safety applications in the presence

of attacks. In addition, all algorithms proposed in this research comply with current protocols and do not require modifications to existing standards.

Acknowledgements

First and foremost, I want to thank GOD, the Almighty, for the completion of this dissertation. This work would not have been possible without the kind support and help of many individuals. I am extending my sincere thanks to all of them.

I would like to express my deepest gratitude to my Major Professor, Axel Krings, for his excellent guidance, caring, patience, and providing me with an excellent atmosphere for doing this research. I am extremely grateful to the rest of my committee: Prof. Robert Rinker, Prof. Fredrick Sheldon, and Prof. Ahmed Abdel-Rahim, for their encouragement and insightful comments. Special thanks goes to Sherif Hussein and Sameh Sorour who are good friends, and always willing to help and give their best suggestions.

Nobody has been more important to me in the pursuit of finishing this research than the members of my family. I would like to thank my parents, whose love and guidance are with me in whatever I pursue. They are the ultimate role models. Most importantly, I wish to thank my loving and supportive wife, Nermeen, and my two wonderful children, Hamza and Laila, who provide unending inspiration, encouraged me and expressed confidence in my abilities when I could only do the opposite. Last, but not least, special and profound thanks to my sisters, Reham and Renal, who offered invaluable support and humor over the years. Words can't express how much I love you all and how grateful I am for your support. Without you seven, I most certainly would not be where I am today.

Dedication

I would like to dedicate this dissertation to praising GOD and asking for his peace and blessing on all of those who contributed to completing this research and making it useful knowledge. This work is also dedicated to:

My father, Sobhy Abdeen,

My mother, Nourelhoda Nasr,

My wife, Nermeen Nassar,

My kids, Hamza and Laila,

My sisters, Reham and Renal,

who have been a constant source of support and encouragement during the challenges of studying and life. I am truly thankful for having you in my life.

Table of Contents

Authorization to Submit Dissertation . . . . . **ii**

Abstract . . . . . **iii**

Acknowledgements . . . . . **v**

Dedication . . . . . **vi**

Table of Contents . . . . . **vii**

List of Figures . . . . . **xi**

List of Tables . . . . . **xiii**

List of Abbreviations . . . . . **xiv**

1 Introduction **1**

1.1 Preliminary Background Information 1

1.2 Research Motivation and Objectives 2

1.3 Summary of Contributions 3

1.4 Dissertation Organization 4

2 Background **5**

2.1 Connected Vehicles 5

2.1.1 Dedicated Short Range Communication (DSRC) 6

2.1.2 DSRC Channel Power Levels 7

2.2 Medium Access Control (MAC) 8

2.3 General Queuing behavior 10

2.4 VANET Applications 11

2.4.1 Basic Safety Message (BSM) 11

2.4.2 DSRC Safety Applications 13

2.4.3 Arada Systems LocoMate OBU and RSU 14

2.5 Malicious Attacks in VANETs . . . 16
2.5.1 Attackers . . . 16
2.5.2 Taxonomy of Jammers . . . 16
Simple Jamming Models . . . 17
Intelligent Jamming Models . . . 17
2.5.3 MAC Misbehavior . . . 17
2.5.4 GPS Spoofing and Bogus Information . . . 18
2.5.5 Sybil Attack . . . 18
2.5.6 Malware and Spam . . . 19
2.6 Reliability of DSRC Safety Applications . . . 19
2.6.1 EEBL Timing Model . . . 19
2.6.2 Reliability of EEBL . . . 19

3 Enhanced Voting Algorithm for Hybrid Jamming Attacks in VANET . . . 21
3.1 Related Work . . . 21
3.2 Observations of Transmission Queue Behavior . . . 24
3.3 A New Hybrid Jammer . . . 25
3.3.1 Hybrid Jammer Properties . . . 26
3.3.2 Jamming Impact on Queuing . . . 27
3.4 Design Concepts . . . 28
3.4.1 Detection of Hybrid Jammer . . . 29
3.4.2 Attack Model . . . 29
3.4.3 Enhanced Voting Algorithm (EVA) . . . 31
3.5 Performance Analysis . . . 33
3.5.1 Setup of Experiments . . . 33
3.5.2 Previous Voting Approaches . . . 34
3.5.3 Performance Evaluation of the EVA . . . 36
3.6 Conclusions . . . 38

4 Sybil Attack Detection Algorithms . . . 39
4.1 Related Research . . . 39
4.1.1 Resources Testing . . . 39

4.1.2 Ranging Methods . . . 40
Received Signal Strength Indicator (RSSI) . . . 40
Time Based Methods . . . 40
4.1.3 Neighbor Information and Collaboration . . . 41
4.1.4 Road-Side Unit (RSU) . . . 41
4.1.5 Cryptography and Authentication . . . 42
4.2 Assumptions and Attack Model . . . 42
4.3 Passive Sybil Detection . . . 44
4.3.1 Notation and Basic Concept Definitions . . . 44
4.3.2 Passive Sybil Detection Algorithm . . . 45
4.3.3 Simulation Setup . . . 48
4.3.4 Simulation Result Analysis . . . 49
4.4 Active Sybil Detection . . . 52
4.4.1 Impact of Forced BSM Queuing . . . 53
4.4.2 Active Sybil Detection Algorithm . . . 54
4.4.3 Field Experiments . . . 56
4.4.4 Experimental Result Analysis . . . 58
4.5 Conclusions . . . 61

5 An Enhanced Active Sybil Detection Algorithm and its Impact on Reliability of Safety Applications in VANET . . . 62
5.1 Fault Model . . . 62
5.2 Enhanced Active Sybil Detection Algorithm (EASDA) . . . 64
5.3 Safety Application Reliability in Presence of EASDA . . . 67
5.4 Experiments and Results . . . 68
5.4.1 Field Experiments . . . 68
5.4.2 Detection Probability . . . 70
5.5 Conclusion . . . 75

6 Conclusions and Future Work . . . 76
6.1 Conclusions . . . 76
6.2 Future Work . . . 78

Bibliography **79**

Appendices **87**

A NS-3 Simulation Functions **87**

A.1 Stationary Position Model 87

A.2 Velocity-Mobility Model 88

A.3 Propagation Models 89

A.4 Determining of the Transmission Range 90

A.5 Main function of Simulation 92

A.6 Sybil Attack Scenario 1 98

A.7 Sybil Attack Scenario 2 99

List of Figures

Figure 2.1 Connected vehicles architecture [modified from [5]] 5
Figure 2.2 Layered architecture for DSRC 6
Figure 2.3 DSRC channels 7
Figure 2.4 EDCA channel access prioritization [14] 9
Figure 2.5 Basic Safety Message (BSM) [12] 12
Figure 2.6 Safety Applications scenarios 13
Figure 2.7 Arada LocoMate OBUs and RSU 15
Figure 2.8 Arada LocoMate OBU block digram 15
Figure 2.9 EEBL BSM Timing Model [43] 20

Figure 3.1 Field experiment scenario [39] 24
Figure 3.2 Field test observations logged by vehicle V3 25
Figure 3.3 Queuing effect for jamming periods of $1, 2, 3$, and $4s$. 28
Figure 3.4 No attack 30
Figure 3.5 Attack causing message queuing 30
Figure 3.6 Enhanced Voting Algorithm for Event e_j 32
Figure 3.7 Performance: 1.5 second jamming with Arada OBU 35
Figure 3.8 Performance: 2 second jamming with Arada OBU. 37
Figure 3.9 Performance: 4.5 second jamming with Arada OBU 38

Figure 4.1 Sample attack scenario. 43
Figure 4.2 Passive Sybil detection algorithm 46
Figure 4.3 Change HV sensitivity S_i. 47
Figure 4.4 NS-3 simulation relative positions of malicious nodes with respect to HV and Sybil nodes 48
Figure 4.5 Determine HV's maximum reception distance 49
Figure 4.6 Relationship of Rx sensitivity and distance - Scenario of Figure 4.4 a). 50

FIGURE 4.7 Relationship of Rx sensitivity and distance - Scenario of Figure 4.4 b). 51
FIGURE 4.8 Actual and expected number of vehicles - Scenario of Figure 4.4 a) 52
FIGURE 4.9 Actual and expected number of vehicles - Scenario of Figure 4.4 b) 53
FIGURE 4.10 Field experiment with a DP duration of 500ms 54
FIGURE 4.11 Cumulative BSM inter-arrival times for a DP of 50ms 55
FIGURE 4.12 Active Sybil detection algorithm 56
FIGURE 4.13 Extreme position of malicious nodes. 58
FIGURE 4.14 Probability of delay detection. 59
FIGURE 4.15 Low-precision scenario F1 60
FIGURE 4.16 High-precision scenario F2 61

FIGURE 5.1 Fault taxonomy 63
FIGURE 5.2 BSM delay due to DP 67
FIGURE 5.3 Relative position of malicious node to Sybil nodes 70
FIGURE 5.4 Probability of delay detection for different τ 72
FIGURE 5.5 Probability of not detecting a delay for DPs with different τ and T 73
FIGURE 5.6 Overhead of DPs with different τ and T 74
FIGURE 5.7 Overhead Ratio ω to achieve $\epsilon = 0.1$ during 2s 75

List of Tables

Table 2.1 DSRC channels peak power limits [4] 8
Table 2.2 Parameter settings for different access classes in IEEE 802.11P [7] 10
Table 2.3 Safety Applications and crash imminent scenarios [5] 14

Table 3.1 Hybrid Jammer Parameters 27
Table 3.2 Field Experiment Configuration Parameters 34

Table 4.1 NS-3 Simulation Parameters 49
Table 4.2 Sybil Detection Field Experiment Parameters 57

Table 5.1 Field Test Parameters 69

List of Abbreviations

ASTM	American Society for Testing and Materials
ACK	Acknowledgment
BSM	Basic Safety Message
BSW	Blind Spot Warning
CA	Certification Authority
CCH	Control Channel
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CLW	Control Loss Warning
CTS	Clear to Send
DSRC	Dedicated Short Range Communication
DoS	Denial of Service
DCF	Distributed Coordination Function
DIFS	Distributed Inter-Frame Space
DNPW	Do Not Pass Warning
DP	Detection Packet
EEBL	Emergency Electronic Brake Lights
EVA	Enhanced Voting Algorithm
EIRP	Effective Isotropic Radiated Power
FIFO	First-in First-out
FSPL	Free Space Path Loss
FCC	Federal Communication Commission
FCW	Forward Collision Warning
GPS	Global Positioning System
HV	Host Vehicle
HJDA	Hybrid Jammer Detection Algorithm
ITS	Intelligent Transportation Systems
IEEE	Institute of Electrical and Electronics Engineer
IMA	Intersection Movement Assist

LIFO	Last-in First-out
LCW	Lane Change Warning
MANET	Mobile Ad-Hoc Networks
MAC	Media Access Control
MIPS	Microprocessor without Interlocked Pipeline Stages
NHTSA	National Highway Traffic Safety Administration
NPD	Newest Packet Drop
OBU	On-Board Unit
OPD	Oldest Packet Drop
PDR	Packet Delivery Ratio
PIFS	PCF Inter-frame Space
PCF	Point Coordination Function
PCI	Peripheral Component Interconnect
PHY	Physical Layer
RSU	Road Side Unit
RV	Remote Vehicle
RSSI	Received Signal Strength Indicator
RF	Radio Frequency
RTS	Request to Send
SAE	Society of Automotive Engineers
SCH	Service Channel
SNR	Signal-to-Noise Ratio
SIFS	Short Inter-frame Space
SDRAM	Synchronous Dynamic Random-Access Memory
USDOT	United States Department of Transportation
V2V	Vehicle-to-Vehicle
V2I	Vehicle-to-Infrastructure
VANET	Vehicular ad hoc Networks
VSC-A	Vehicle Safety Communications - Applications
WLAN	Wireless Local Area Network

CHAPTER 1

Introduction

During the last three decades there have been many innovations in vehicles in terms of fuel efficiency and navigation, but there has not been much change in terms of road safety. Road travel is still considered to be quite hazardous as the mistake of a single person can have catastrophic results, especially at high speeds. According to the National Highway Traffic Safety Administration (NHTSA), there were 6.3 million police-reported crashes in 2015, and the number of fatalities from vehicle crashes accounted for 35,092 deaths in the U.S., in addition to 2.44 million injuries [1].

The automotive industry has been working actively for years to put different sensors in cars and connect them to an on-board computer [2]. The advancement in telecommunications have allowed vehicles to be connected to each other through wireless technologies to enable them to share sensor information and cooperate.

A group of communicating vehicles can form a de-centralized network, referred to as a Vehicular Ad Hoc Network (VANET). This paradigm allowed a wide range of safety applications to be developed, with the goal to improve road safety, thereby reducing accidents. It is estimated by the United States Department of Transportation (USDOT) that safety applications can prevent up to 82% of all crashes in the United States that involve unimpaired drivers, potentially saving thousands of lives and billions of dollars [3]. However, the deployment of VANETs requires well-defined technologies in order to ensure safe, stable, and reliable system operation.

1.1 Preliminary background information

Intelligent Transportation Systems (ITS) are utilizing technology to increase traffic safety and provide environmental benefits [4]. A variety of technologies have been applied, ranging from basic traffic management systems, such as car navigation and traffic signal control systems, to advanced technology aiming to increase traffic safety. In the context of ITS *connected vehicles* represent one of the newest concepts, where

technologies, services, and applications allow wireless communication between vehicles and between vehicles and the roadside infrastructure.

VANET is somewhat similar to Mobile Ad Hoc Network (MANET), but it is intended for quick message exchanges and connections that may change rapidly. Vehicles communicate with each other via radios (wirelessly) and sometimes with the help of radio-enabled fixed road side infrastructure. There are numerous VANET applications that have been proposed, safety applications being the most important ones [5]. The applications rely on beacon messages that are sent periodically by all vehicles to exchange data, which is used by safety applications to issue driver advisories and warnings.

1.2 RESEARCH MOTIVATION AND OBJECTIVES

Communication in VANET, which is the basis for safety applications, may be subjected to the full spectrum of security concerns associated with such technologies. Thus it is susceptible to different types of attacks ranging from simple corruption of communication, e.g., using jamming, to sophisticated attacks, e.g., Sybil attacks, where a node pretends to be multiple fake nodes. In fact, safety applications can be manipulated by attackers in a way that may lead them to make wrong decisions, which in turn could result in accidents. Furthermore, since these safety applications operate in a critical infrastructure, where failure could result in injury and loss of life, safety and reliability are crucial. Any failure, may it be the result of benign reasons or malicious act, could result in the public's loss of confidence in the underlying technologies.

As security is perhaps the greatest challenge facing the deployment of VANETs, this dissertation focuses on developing solutions to increase reliability of safety applications in the presence of faults and attacks. The solutions should be resilient to malicious acts in VANET. Therefore, research questions such as the following arise:

1. How should one deal with false data injected by rouge nodes?

2. How can one keep illegitimate (rogue - malicious) nodes out of the network?
3. How can one detect rogue nodes that might be already in the network?
4. How can we accomplish these goals without deviating from standards, or are changes necessary?

Any solutions should be scalable and real-time feasible.

1.3 SUMMARY OF CONTRIBUTIONS

The main contributions of this dissertation are summarized next.

1. During field tests related to the study of the impact of deceptive jamming on vehicle communications, we observed unexpected excessive queuing behavior in the on-board units of vehicles. The observations motivated the derivation of a new hybrid jammer and its detection mechanism. This jammer exposed queuing behavior that can be exploited by attackers to cause safety application failure, and it can make innocent nodes appear misbehaving, as if excessively using the medium. A new detection algorithm is introduced that can not only detect the hybrid jammer, but can also distinguish between misbehaving nodes and nodes that are impacted by the jammer. Current Packet Delivery Ratio (PDR)-based detection methods fail to detect this new jammer.
2. The impact of the hybrid jammer on safety applications using voting schemes was studied and an Enhanced Voting-based Algorithm (EVA) was derived. The EVA operates in two stages. The first stage uses specific metrics that can differentiate between misbehaving nodes and nodes that are impacted by the jammer. The second stage considers delayed beacon messages coming from victim nodes, which in turn directly affect decisions of safety applications. The EVA is capable of significantly reducing the application's decision times compared with previous algorithms, thereby improving application reliability, and thus safety.

3. A Sybil attack detection algorithm is introduced to guide Dedicated Short Range Communications (DSRC) safety applications to reject malicious nodes. The algorithm is an active solution that can locate Sybil nodes using short detection packets without adding special hardware or information exchanges. In addition, unlike in previous detection approaches, it is capable of Sybil detection even in dynamic power environments. In the context of safety critical applications, this will help avoid accidents by rejecting values from malicious nodes, which is of great importance for solutions based on voting schemes.

4. The impact of the frequency and duration of detection packets on Sybil node detection is analyzed. An enhanced active Sybil detection algorithm using these two metrics is presented and its impact on DSRC safety application reliability is shown.

1.4 DISSERTATION ORGANIZATION

The remainder of this dissertation will be organized as follows: Chapter 2 presents important background information. A New Hybrid Jammer and the Enhanced Voting based Algorithm is discussed in details in Chapter 3. Novel Sybil attack detection algorithms in VANET are presented in Chapter 4. An Enhanced Active Sybil Detection Algorithm and its impact on the reliability of safety applications in VANET is discussed in Chapter 5. Finally, Chapter 6 concludes the paper.

CHAPTER 2

Background

The fast development of ITS have allowed vehicles to be equipped with wireless communication capabilities to be connected and sharing information about road. In this chapter background information of the key technologies used in this research is presented.

2.1 CONNECTED VEHICLES

Figure 2.1 refers to *connected vehicles*, which are vehicles that use communication technology to communicate with other vehicles or with the roadside infrastructure. This however requires each vehicle to be equipped with an On Board Unit (OBU)

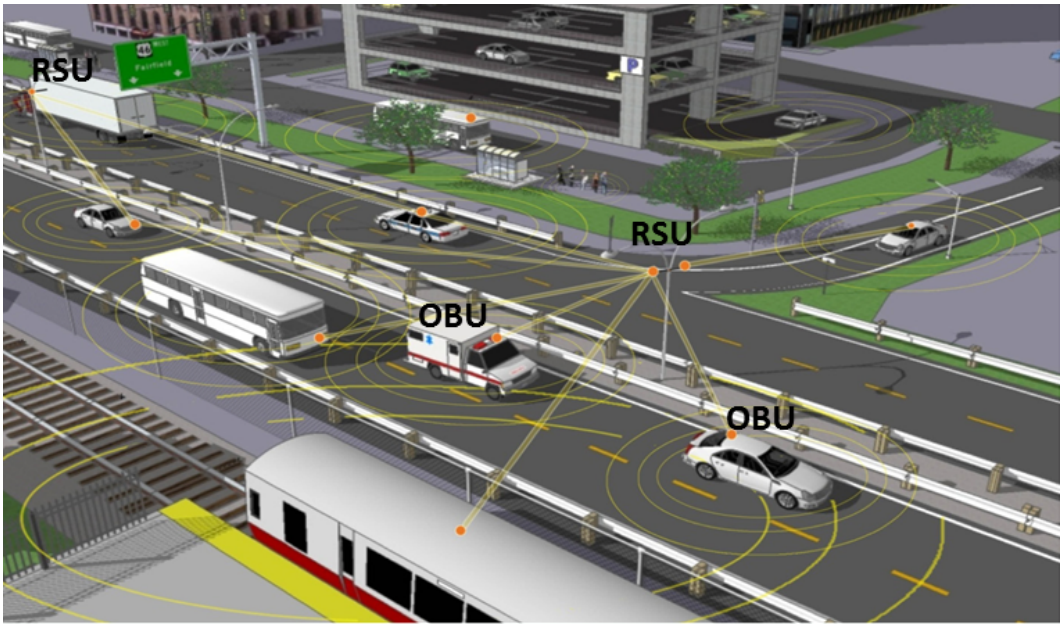


FIGURE 2.1: Connected vehicles architecture [modified from [5]]

and the infrastructure, such as a traffic intersection, with a Road Side Unit (RSU), to allow Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication. These terms V2V and V2I could also combined to V2X. Vehicles are equipped with standard OBUs with the same antenna properties, such as gains and sensitivity. The

OBU's sensitivity is the minimum input signal required at its receiver's antenna to produce a minimum required Signal to Noise Ratio (SNR) at the output of the receiver.

2.1.1 *Dedicated Short Range Communication (DSRC)*

The overall architecture for DSRC is shown in Figure 2.2. A set of industry standards has been published to cover each layer of the architecture and to address proper interoperability [6, 7, 8, 9, 10, 11, 12].

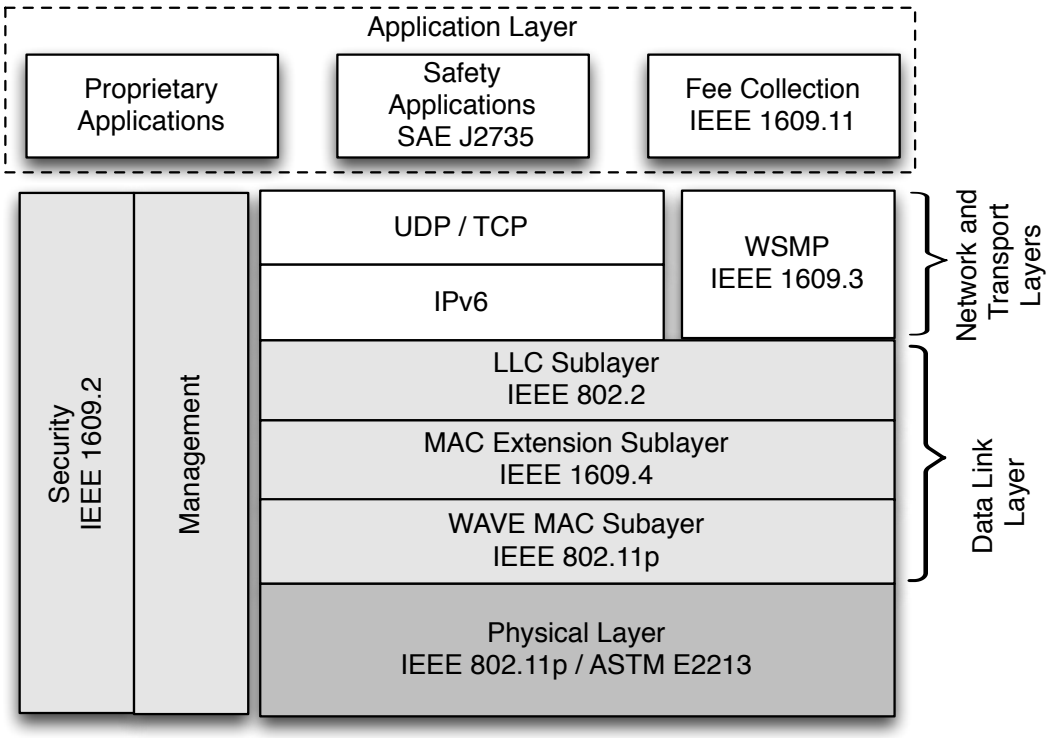


FIGURE 2.2: Layered architecture for DSRC

DSRC provides V2V and V2I communication. The Federal Communications Commission (FCC) has licensed the use of 75 MHz of bandwidth at 5.9 GHz (5.850-5.925 GHz) for DSRC services. This bandwidth is divided into seven channels, each having 10 MHz of bandwidth [4]. The seven channels are composed of one Control Channel (denoted by CH 178), and six Service Channels (denoted by CH 172, 174, 176, 180, 182, and 184). The control channel is reserved for carrying high-priority short messages or management data, while other data are transmitted on the service

channels. The pair of channels (CH 174 and 176, and CH 180 and 182) can be combined to form a single 20-MHz channel, CH 175 and CH 181 respectively. The most important channel is Safety Channel CH 172, dedicated to V2V public safety communications. The remaining 5 MHz is reserved as the guard band, as can be seen in Figure 2.3.

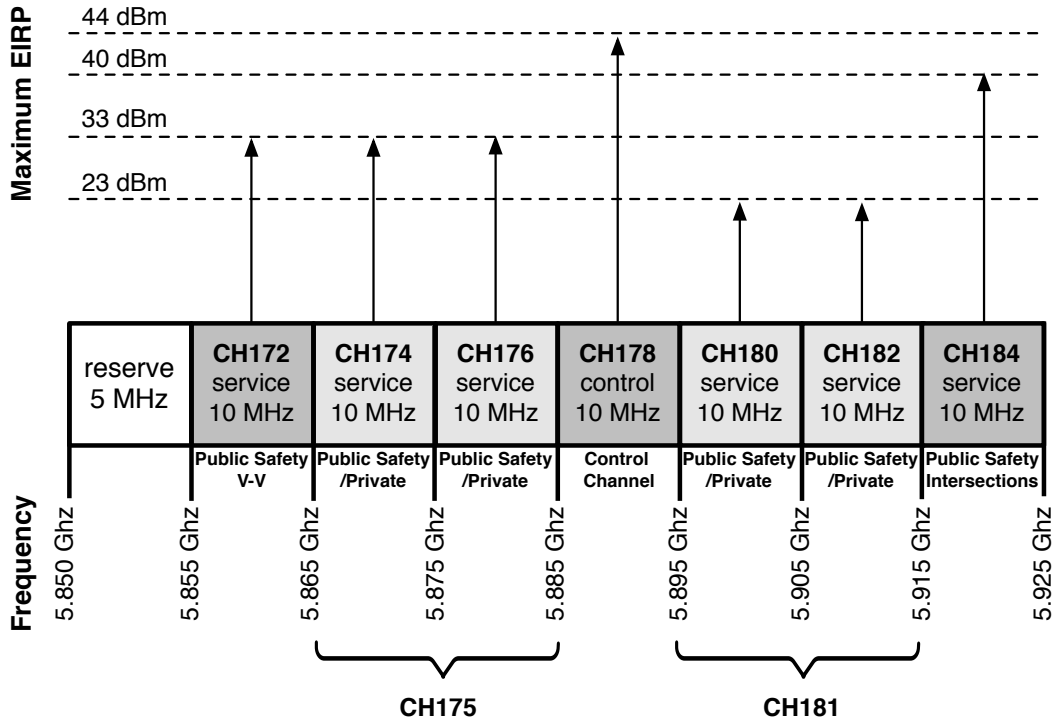


FIGURE 2.3: DSRC channels

2.1.2 *DSRC Channel Power Levels*

Different channels have different power levels and limitations, which should be considered in the safety application design phase. Table 2.1 summarizes power levels for different DSRC channels. According to [4, 6], the maximum transmitted power over the DSRC frequency band of operations is limited to no more than 750 mW (28.8 dBm). Under the assumption that the signal is radiated equally in all directions, the Effective Isotropic Radiated Power (EIRP), which is the amount of power that a theoretical isotropic antenna would emit to produce the peak power

TABLE 2.1: DSRC channels peak power limits [4]

CH	Operation	Public Safety		Private	
		Input Power (dBm)	EIRP (dBm)	Input Power (dBm)	EIRP (dBm)
172	RSU	28.8	33	28.8	33
	OBU	28.8	33	28.8	33
174	RSU	28.8	33	28.8	33
	OBU	28.8	33	28.8	33
175	RSU	10	23	10	23
	OBU	10	23	10	23
176	RSU	28.8	33	28.8	33
	OBU	28.8	33	28.8	33
178	RSU	28.8	44.8	28.8	33
	OBU	28.8	44.8	28.8	33
180	RSU	10	23	10	23
	OBU	NA	NA	NA	NA
181	RSU	10	23	10	23
	OBU	NA	NA	NA	NA
182	RSU	10	23	10	23
	OBU	NA	NA	NA	NA
184	RSU	28.8	40	28.8	33
	OBU	28.8	40	28.8	33

density observed in the direction of maximum antenna gain [13], is not greater than 30 W (44.8 dBm), and is determined using

$$EIRP_{log} = P_T - L_c + G_a \tag{2.1}$$

where P_T is the transmission power in dBm, L_C is the signal loss in dB, and G_a is the antenna gain in dBi, relative to an isotropic reference antenna. This means that implementations should address the channel and power limits defined in the ASTM-DSRC standard [6].

2.2 MEDIUM ACCESS CONTROL (MAC)

The access rules to the medium are defined in standard IEEE 802.11p [7], which was proposed to be used in rapidly changing environments where very short communication durations are required. The standard employs the Enhanced Distributed Channel Access (EDCA) contention-based channel access as the Medium Access

Control (MAC) layer protocol. EDCA utilizes Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). When a node wants to send a packet, it senses the medium first, and if it is free for an Arbitration Interframe Space (AIFS), the node selects a random backoff time to delay the transmission. Figure 2.4 depicts the timing related to channel access for different interframe spacings.

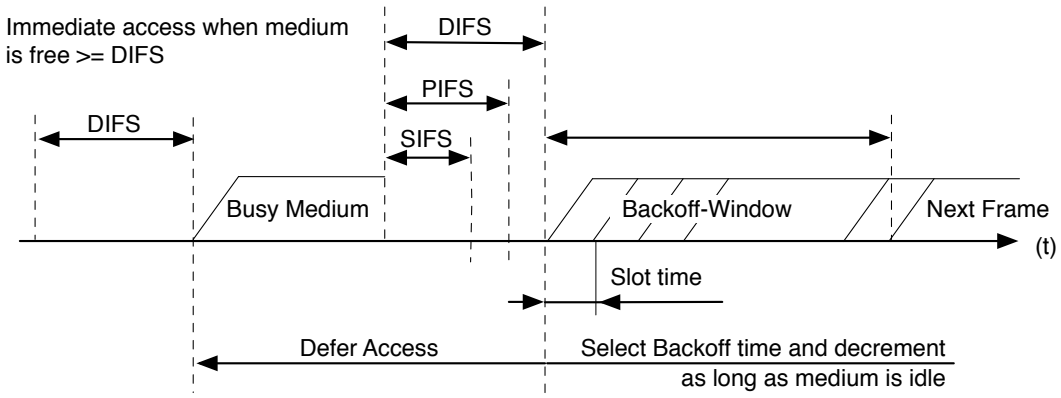


FIGURE 2.4: EDCA channel access prioritization [14]

The backoff procedure functions as follows:

1. The node selects a random backoff time uniformly from the Contention Window CW defined as $[0, CW + 1]$, where the initial CW is equal to a predetermined CW_{min}.

2. The CW value increases, i.e., it doubles, if the subsequent transmission attempt fails, until it reaches a predetermined CW_{max}.

3. The backoff value decreases only when the node senses the medium as free.

4. When the backoff value reaches 0, the node will send the packet immediately.

EDCA utilizes four different access categories (ACs) with different priorities in order to ensure that high priority messages will be exchanged timely and reliably, even in dense traffic scenarios. These four traffic categories are: Background traffic (BK or AC0), Best Effort traffic (BE or AC1), Video traffic (VI or AC2) and Voice traffic (VO or AC3).

Table 2.2 shows different AIFS numbers and CW values chosen for each of these four categories. The first column contains the four ACs ordered from lowest

TABLE 2.2: Parameter settings for different access classes in IEEE 802.11P [7]

AC	CW_{min}	CW_{max}	AIFSN
BK (AC0)	CW_{min}	CW_{max}	9
BE (AC1)	CW_{min}	CW_{max}	6
VI (AC2)	$(CW_{min})/2$-1	CW_{min}	3
VO (AC3)	$(CW_{min})/4$-1	$(CW_{min})/2$-1	2

to highest priority. The second and the third columns show the corresponding minimum and maximum contention windows. Finally, the last column holds the Arbitration Interframe Space Number (AIFSN) used.

2.3 GENERAL QUEUING BEHAVIOR

After creation of a packet to be sent, it is placed in the First-in First-out (FIFO) transmission queue [7, 11]. Once the medium is free, the packet is taken from the queue to be transmitted. Channel congestion is expected in high traffic situations, where a node may not be able to send its packets, resulting in message queuing for that node. This could go on until the capacity of the transmission queue overflows, in which case packets are dropped. The performance of sending packets can be affected by the transmission queue size and the scheduling strategy used.

With respect to timeliness, the size of the queue can effect the freshness of the packet. A large queue can hold more packets and one can expect that fewer packets will be dropped. However, the longer a packet is queued, the more outdated its information becomes, which could make the packet useless. In the context to buffering and scheduling, queuing issue were discussed in [15]. Two queuing strategies were investigated for use in the MAC layer. The first, Newest Packet Drop (NPD) or tail-drop queuing, means that once the queue is full, the new arriving packet will be dropped. The second, Oldest Packet Drop (OPD), also known as head-drop queuing, implies that when a packet arrives at a full queue, the oldest packet will be placed with the new one arrives.

2.4 VANET APPLICATIONS

VANET applications can be classified into three categories. The first category is convenience applications, which mainly deal with traffic management aiming to enhance traffic efficiency. The second category has commercial applications that provide drivers with entertainment and services, such as web access, streaming audio and video [16, 17]. The third category contains safety applications, which are considered one of the most important applications in VANET proposed by [18].

2.4.1 *Basic Safety Message (BSM)*

The Basic Safety Message (BSM), which is a beacon message broadcast periodically every 100ms on CH172, is the most important message used in a variety of DSRC safety applications to exchange information about the status of the vehicle [12]. A BSM consists of two parts. The first part is mandatory and contains data included in every BSM. The second part is optional. It is transmitted less frequently and includes additional information for certain applications.

Figure 2.5 shows the Basic Safety Message (BSM) message format. The first part of the BSM consists of 39 bytes that contain frequently used fields. *DSRC_MessageID* is used to identify the message type, which helps the receiving application in interpreting the remaining message bytes. *MsgCount* shows the number of messages that were sent by the same vehicle with the same *MessageID*. *TemporaryID* is used to identify the local vehicles that are interacting during an encounter. This ID periodically changes to ensure the vehicle's anonymity. *DSecond* provides current timing information consisting of an integer value representing the milliseconds within a minute. The mandatory part of a BSM also includes GPS related information, e.g., *Latitude*, *Longitude*, *Elevation* and *PositionalAccuracy*. Motion and control information fields are represented next. *TransmissionAndSpeed* states the current speed combined with a value indicating the vehicle's transmission state. *Heading* provides the current heading and orientation of the vehicle. *SteeringwheelAngel* indicates the rate of change of the steering wheel angel in either direction. *AccelerationSet4Way* provides acceleration values in three orthogonal directions and yaw rotation rates.

```
BasicSafetyMessageVerbose ::= SEQUENCE {
   -- Part I, sent at all times
   msgID       DSRCmsgID,              -- App ID value, 1 byte

   msgCnt      MsgCount,               -- 1 byte
   id          TemporaryID,            -- 4 bytes
   secMark     DSecond,                -- 2 bytes
   -- pos      PositionLocal3D,
   lat         Latitude,               -- 4 bytes
   long        Longitude,              -- 4 bytes
   elev        Elevation,              -- 2 bytes
   accuracy    PositionalAccuracy,     -- 4 bytes

   -- motion   Motion,
   speed       TransmissionAndSpeed,   -- 2 bytes
   heading     Heading,                -- 2 bytes
   angle       SteeringWheelAngle,     -- 1 bytes
   accelSet    AccelerationSet4Way,    -- 7 bytes

   -- control  Control,
   brakes      BrakeSystemStatus,      -- 2 bytes

   -- basic    VehicleBasic,
   size        VehicleSize,            -- 3 bytes

   -- Part II, sent as required
   -- Part II,
   safetyExt   VehicleSafetyExtension OPTIONAL,
   status      VehicleStatus          OPTIONAL,
   ... -- # LOCAL_CONTENT
   }
```

FIGURE 2.5: Basic Safety Message (BSM) [12]

BrakeSystemStatus provides information about the current brake system status, e.g., brake usage, anti-lock brake status, and auxiliary brake status. Finally, *VehicleSize* indicates the vehicle's length and width.

The optional part contains fields application specific information. For example, *VehicleSafetyExtention* is used to send various additional details about the vehicle, such as Event Flags, Path History and Path Prediction. *VehicleStatus* contains information to relate specific items of the vehicle's status. It is typically used in data event snapshots, which are gathered and periodically reported to an RSU.

2.4.2 *DSRC Safety Applications*

Various types of DSRC safety applications, which aim to enhance safety by notifying drivers of potential hazards or accidents, have been presented in [5]. The safety applications rely on BSMs that are periodically sent by each vehicle's OBU every 100ms. The information contained in BSMs received from surrounding vehicles is used to alert drivers about impending dangers. The USDOT evaluated pre-crash scenarios based on the National Automotive Sampling System (NASS) crash database in order to provide a list of potential crash imminent safety scenarios [19]. This analysis resulted in the identification and selection of the following safety applications described in Figure 2.6 to be developed and implemented as part of the VSC-A program:

Forward Collision Warning (FCW), depicted in Figure 2.6 a), is intended to warn the driver of the Host Vehicle (HV) in case of an impending rear-end collision with a Remote Vehicle (RV) ahead in the same lane and direction of travel. FCW helps drivers avoid rear-end vehicle collisions in the forward path of travel.

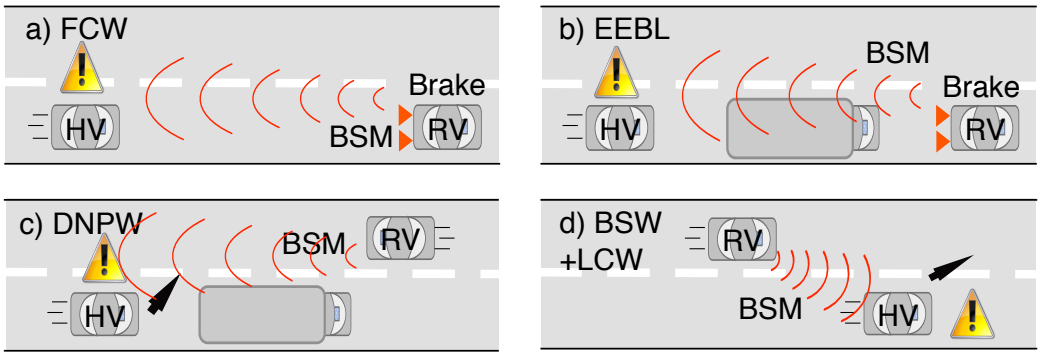


FIGURE 2.6: Safety Applications scenarios

Figure 2.6 b) describes *Emergency Electronic Brake Lights* (EEBL), which enables vehicles to broadcast an emergency brake event to surrounding vehicles. Upon receiving such event, the HV determines the relevance of the event and provides a warning to the driver. EBBL is useful in situations of low visibility, e.g., due to fog, or when other vehicles block the view of the HV.

Do Not Pass Warning (DNPW), shown in Figure 2.6 c), alerts the driver of the HV during a passing maneuver that a slower moving vehicle cannot be safely passed

using the passing zone, e.g., as it will be soon occupied by a vehicle traveling in the opposite direction.

Blind Spot Warning + Lane Change Warning (BSW+LCW), shown in Figure 2.6 d), is designed to warn the driver of the HV who attempts to change lane, but this intended lane is occupied by another vehicle positioned in a blind-spot zone and traveling in the same direction.

Intersection Movement Assist (IMA) is intended to warn the driver of an HV when it is not safe to enter an intersection due to high collision probability with other RVs crossing this intersection.

Table 2.3 illustrates the relationship/mapping between the crash scenarios identified by the USDOT and the list of high impact safety applications described above.

TABLE 2.3: Safety Applications and crash imminent scenarios [5]

	Crash Scenarios / Safety Applications	**EEBL**	**FCW**	**BSW**	**DNPW**	**IMA**	**CLW**
1	Lead Vehicle Stopped		✓				
2	Control Loss without Prior Vehicle Action						✓
3	Vehicle(s) Turning at Non-Signalized Junctions					✓	
4	Straight Crossing Paths at Non-Signalized Junctions					✓	
5	Lead Vehicle Decelerating	✓	✓				
6	Vehicle(s) Changing Lanes - Same Direction			✓	✓		
7	Vehicle(s) Making Maneuver - Opposite Direction					✓	

2.4.3 *Arada Systems LocoMate OBU and RSU*

The Arada LocoMate devices shown in Figure 2.7 are examples of commercial OBUs and RSU. These devices enable wireless connectivity V2V and V2I in vehicular environments [20]. The LocoMate OBU operates within the physical specification of the FCC amendment and the ASTM E2213 standard [4, 6]. Moreover, it provides safety and data services to the vehicle users by communicating BSMs per the SAE J2735 standard [12]. The OBU is Linux/Unix compatible and comes with a Software Development Kit (SDK) with C libraries. It has an integrated GPS device with external RF antenna with an accuracy of up to 1m, which provides location information such as the longitude and latitude of the vehicle.

Figure 2.8 shows the OBU's block digram. According to the manufacturer's user manual [20], the OBU utilizes a 680MHz MIPS processor and 16MB of flash memory,

FIGURE 2.7: Arada LocoMate OBUs and RSU

in addition to a 64MB of SDRAM. It has a Gigabit Ethernet interface and an Atheros AR5414 based WLAN Mini PCI. It should be noted that we used these devices during

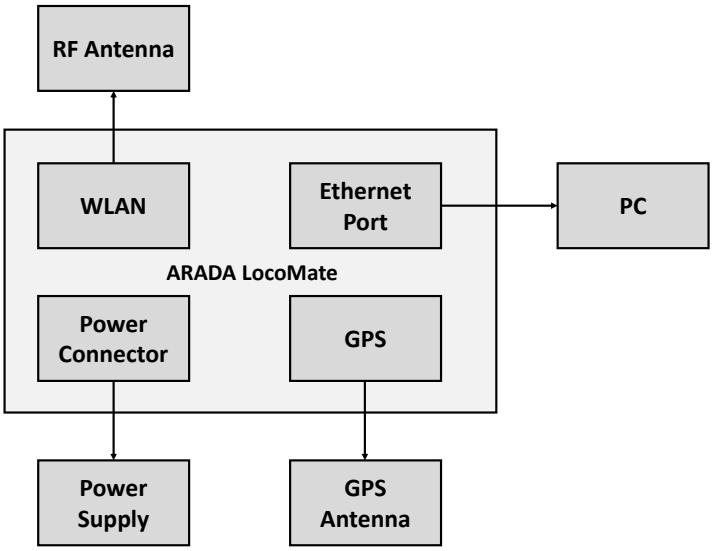


FIGURE 2.8: Arada LocoMate OBU block digram

flied experiments. Each OBU was connected to the vehicle's power supply, and the GPS antenna was placed on the top of the vehicle. The OBUs were mounted on the roof of the vehicles with upward antenna orientation. In order to allow real-time monitoring they were connected to computers in the vehicles via Ethernet.

2.5 MALICIOUS ATTACKS IN VANETS

As safety applications use wireless communication, they are potentially subjected to all security challenges associated with this communication paradigm. The aim of malicious attack could be either to cause safety applications to fail [21, 43], or to get right of way by giving the illusion of congestion [22] in active traffic management. Some attacks aim at reducing the SNR by emitting radio signals that interfere with the communication [23]. Other attacks could disobey medium access rules. They could insert bogus packets, the worst of this is sending packets with false information like Sybil attacks [24]. Understanding the types of security threats and attackers is very important to design any security solution for VANETs.

2.5.1 *Attackers*

According to [25], attackers are classified into four basic categories based on the scope and behavior of attacks. *Insiders vs. outsiders*: insiders are authenticated members in a network, while outsiders are intruders with less capabilities. *Malicious vs. rational*: malicious attackers intend to cause accidents but do not personally benefit from this attack, while rational attackers have specific goals. *Active vs. passive*: active attackers send fake or manipulated messages, whereas passive attackers sniff the network to collect information for future attacks. *Local vs. extended*: the scope of a local attacker is of limited range, whereas extended attackers target the larger network.

2.5.2 *Taxonomy of Jammers*

Wireless jamming is a common attack in wireless communication. It can be defined as an act of transmitting radio signals in order to interfere with communication and block legitimate nodes from accessing the medium. The goal of jammers may be twofold, 1) to interfere with wireless communication to decrease the SNR, thus making reception unreliable or impossible and potentially destroy network packets, or 2) prevent nodes from gaining access to the medium. Jammers may have different properties, including ease of detectability, power usage, sophistication with respect

to protocol awareness and level of Denial of Service (DoS) [23, 26]. Jamming models can be divided into two main categories: 1) simple jamming models and 2) intelligent jamming models.

Simple Jamming Models — Simple jammers were discussed in detail in [26]. The first jamming model in this category is the *Constant Jammer*, which emits a constant stream of random data that does not follow the MAC layer protocol. As a result, the medium appears to be busy, thus blocking legitimate nodes from access. However, it may also result in corruption of ongoing packets. The *Deceptive Jammer* is the second jamming model. This type of jammer does not follow the channel access protocol by continually injecting a stream of what appears to be valid packets without any gaps between them. The third jamming model, *Random Jammer*, switches randomly between periods of jamming and sleeping. During the jamming period its behavior resembles that of a constant jammer. *Reactive Jammer* is the last jamming model in this category. It senses the medium for ongoing communication, and when it senses a packet transmission it emits a radio signal that collides with the packet, thus corrupting it.

Intelligent Jamming Models — This type of jammer interferes with communication between nodes, thereby corrupting packets sent by legitimate nodes. It is also called protocol-aware jammer, and has the capability of corrupting specific packets. It may target control packets, such as *Request to Send (RTS), Clear to Send (CTS)* or *Aknowlagment (ACK)*, but could also target *DATA* packets, as described in [23].

2.5.3 *MAC Misbehavior*

The MAC protocol described in Subsection 2.2 assumes that any node intending to access the medium obeys the MAC rules. Misbehaving nodes are those nodes that violate the MAC access rules. According to [27], there are two types of MAC layer misbehavior. *Selfish misbehavior* implies that nodes want to gain an unfair advantage in transmitting their packets [28, 29]. *Malicious misbehavior* implies that nodes aim to

prevent other legitimate nodes from transmitting packets [26, 30]. This misbehavior can lead to DoS attacks. A considerable amount of research has focused on detecting selfish behavior, including [31, 32]. Detection mechanisms for malicious misbehavior have been presented in [30, 33].

2.5.4 *GPS Spoofing and Bogus Information*

In *GPS spoofing* the attacker attempts to deceive a GPS receiver by broadcasting incorrect GPS signals, stronger than those generated by genuine satellites. The goal is to fool drivers by providing false locations.

Bogus information could be introduced by outsiders (intruders) or by legitimate users. The attacker is capable of injecting and disseminating faulty information in the network for personal advantage that impacts the decisions of other drivers. For instance, a false message can be transmitted announcing "Heavy traffic conditions" to convince other vehicles to take other roads to the benefit the attacking vehicle.

2.5.5 *Sybil Attack*

The Sybil attack was described in [36], where one node, called the *Malicious/Rogue* node, pretends to be multiple nodes by impersonating their identities using stolen or forged IDs. In other words, the attacker simulates several nodes in the network. These simulated nodes are called *Sybil* nodes. The attacker is an insider, rational, and active. A Sybil attack might be launched with different goals. One of the goals may be to give the illusion of a traffic jam. However, the attacker may be more harmful by trying to provoke fake events, e.g., a collision, to force safety applications that use voting schemes to make wrong decisions [38, 39, 40, 41]. The attacker attempts to stack wrong values, sent by Sybil nodes, in the voting sets of safety applications to out-vote the correct values, thus causing safety applications to fail. The misbehaving detection techniques presented in [30, 33] fail to detect Sybil nodes because they obey medium access rules.

2.5.6 *Malware and Spam*

These attacks, such as viruses and spam, may cause severe disturbance in normal VANET behavior [34]. This type of attack is invoked by malicious insiders rather than outsider attackers. Specifically, an attacker broadcasts spam messages in the network to consume the bandwidth, thus increasing the transmission latency. On the other hand, malware is similar to a virus that inhibits normal operation of the VANET. Networks get infected with this kind of attack during software updates of OBUs or RSUs.

2.6 RELIABILITY OF DSRC SAFETY APPLICATIONS

The reliability of DSRC safety applications is related to the probability of a vehicle receiving BSMs in time. The standard definition of reliability R(t) is "*the probability that a system is working to specifications during the entire time interval* $[0, t]$" [42]. Reliability of safety applications will be introduced using the example of EEBL in the context of its timing model.

2.6.1 *EEBL Timing Model*

The timing model related to EEBL is shown in Figure 2.9. Upon recognition of a hazard the driver of the RV is assumed to brake hard at time t_{brake}. During the time interval T_{brake} the RV broadcasts this braking event e in its BSMs to the surrounding vehicles. Note that we use lower case t to denote instances of time and upper case T for time intervals. The EEBL application running on the HV uses the received BSMs indicating event e to alert the driver of the HV. This alert has to be issued early enough to allow the driver to react, i.e., no later than t_{react}. Typical reaction times T_{react} have been recorded within 0.9 to 1.2 seconds [42].

2.6.2 *Reliability of EEBL*

The reliability of EEBL is conditioned on the reception of these BSMs and making the correct decision in the proper time. Assume that the distance d between the HV and

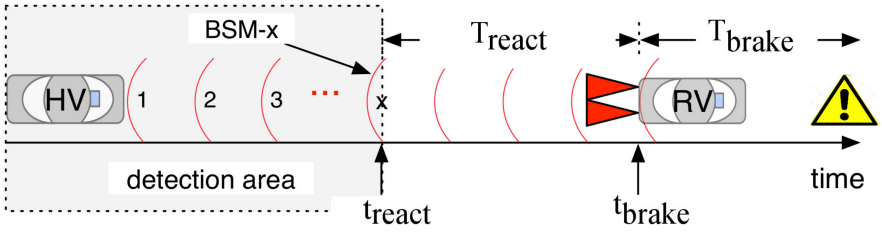


FIGURE 2.9: EEBL BSM Timing Model [43]

RV is equivalent to T_d seconds. Given that BSMs are spaced 0.1s in time, this distance accounts for $b = \lfloor T_d/0.1 \rfloor$ BSMs. However, one can only consider those BSMs that are received at or before t_{react}. Then reaction time accounts for $r = \lfloor T_{react}/0.1 \rfloor$ BSMs. Therefore, the application reliability is the probability of receiving at least one BSM_i, for $i = 1, ..., x$, before it is too late to react, where $x = b - r$.

The application fails if no BSM was received, thereby leaving the HV unaware of the occurrence of the event. If one assumes that the reliability of one BSM is independent of that of another BSM, and using the unreliability $Q(t) = 1 - R(t)$, then the probability that not a single BSM_i is received is [43]

$$Q(t) = \prod_{i=1}^{x} Q_i(t_i) \tag{2.2}$$

where $Q_i(t_i)$ is the probability that BSM_i is not received and t_i is the time it should have been received.

CHAPTER 3

Enhanced Voting Algorithm for Hybrid Jamming Attacks in VANET

In this chapter we address safety application reliability in the presence of jamming, an unavoidable attack in wireless technologies. Specifically, a new hybrid jammer is introduced, which combines the properties of constant and deceptive jammers, in addition to characteristics resembling random jammers. It is shown that this simple-to-implement jammer can manipulate transmitting nodes in a way that can cause safety applications to fail. Moreover, it makes innocent nodes appear as misbehaving on the medium. All this can be done without destroying messages. Results from lab experiments with commercial DSRC equipment, as well as findings during field experiments are presented to demonstrate its effectiveness. Consequently, a detection algorithm as a mitigation strategy for this new jammer is introduced. Next, the impact of hybrid jamming on voting-based approaches is investigated, and an EVA is derived, which is capable of overcoming deficiencies of previous algorithms. The EVA improves decision times without any alterations of existing protocols and standards. Finally, experimental results validate that the new algorithm is superior in terms of time required to make decisions and reliability compared to previous work reported in the next section.

3.1 RELATED WORK

The reliability of safety applications is paramount. For safety applications subjected to malicious act capable of tricking them into deriving incorrect decisions, special mechanisms are needed. Research has addressed this by using redundant BSMs received from nearby vehicles capable of witnessing an event. These vehicles are said to be located in the *detection zone* [38]. Upon detection of an event, each HV starts collecting the BSMs received from vehicles in the detection zone to construct a voting set. Voting-based solutions have been presented in the literature [38, 39, 44]. These

approaches differ in the way a voting set is constructed, e.g., based on the freshness of messages, and the size of the voting set. The latter determines the *voting threshold*. The final decision is based on this threshold, e.g., by applying majority voting.

The challenge is how to select the correct threshold in a trade-off space between speed and robustness of the voting decision. Selecting a low threshold allows decisions to be made fast, however, it may increase the probability of making wrong decisions. On the other hand, selecting a high threshold makes robust decisions, but results in higher latency. Different strategies have been used to define the threshold as *static* or *dynamic* [38, 40]. A static threshold is set a priori, e.g., during manufacturing of the vehicle, while dynamic thresholds change based on neighborhood density and criticality of the event.

Algorithms based on voting can be classified into two categories. The first category consists of voting algorithms using new message architectures based on authentic consensus, namely authentication and verification, of each vehicle. The second category consists of voting algorithms relying on configuring the voting set based on factors like message freshness and thresholds.

An example of the first category, described in [40], considers a Proof-of-Relevance (PoR), which is generated by vehicles collecting digital endorsements from other witnesses of an event. Its scheme consists of three phases. 1) Report generation: This includes location, type and time of an event. 2) Signature collection: It is the key procedure in this scheme. In this phase all vehicles that detected the event will participate in the signature collection protocol until enough signatures are collected. 3) Report verification: Each vehicle that received the event report will examine whether there are enough signatures or not. If there are enough signatures, each vehicle will start validating signatures to check for incorrect signatures. Once enough correct signatures are observed, a decision is made. However, such an approach requires additional communication, thereby adding overhead, but more importantly, they require a modification of the standards.

For the aforementioned reasons we will consider approaches of the second category using voting set configuration as discussed in [38, 39, 41, 44]. In [44] the authors proposed four static decision methods, which are based on voting algorithms that

use plausibility checks in order to take the correct decision in the presence of value faults. These decision methods are: *Freshest Message*, which only consider the most recent messages, *Majority Wins*, which execute local voting over all distinct messages, *Majority of Freshest X*, which combines the previous two methods considering only the recent X distinct messages, and *Majority of Freshest X with Threshold*, which is simply an extension of the previous method in addition to checking if the distinct messages received so far exceed a certain threshold or not. However, their work did not specifically state the *time to live* for messages and does not explicitly state a way for the calculation of the thresholds.

In [38] the authors proposed a dynamic criticality threshold based on the *Majority of Freshest X with Threshold* scheme of [44], where consensus parameters and threshold are depending on neighborhood vehicle density and criticality of the event. There are two strategies for making a decision. As mentioned before, for critical events a compromise space exists between fast and robust decisions. The more critical the event, the fewer messages should be needed for fast decisions. However robust decisions require more messages.

The authors in [41] proposed an adaptive decision making method in order to improve the accuracy and time efficiency of decision-making. It aims to take a decision as soon as possible once the amount of received opinions are greater than a threshold or when the time delay between the first received message and the current received message exceed a maximum delay.

The adaptive threshold algorithm proposed in [39] considered the Majority of Freshest X with dynamic Threshold [38] for an adaptive threshold algorithm that provided higher resilience against certain types of jamming.

None of the previous research is suitable for dealing with the jammer model used in this research, since this jammer model affects the freshness of messages as will discussed in the next sections.

3.2 OBSERVATIONS OF TRANSMISSION QUEUE BEHAVIOR

As mentioned earlier in Section 2.3, after a BSM is created it is placed in the transmission queue, waiting to be sent once the medium is free. Assuming no misbehaving nodes, the normal BSM generation rate of 10 BSM/s, and that the vehicle density is not causing saturation of the medium, BSMs are unlikely to be queued. However, during field tests related to the study of the impact of deceptive jamming on V2V communications, we observed excessive queuing behavior. Figure 3.1 depicts the position of vehicles during the experiments. In the field test, three OBU-equipped vehicles, RV, HV, and V3 passed a stationary deceptive jammer at a speed of about 35mph.

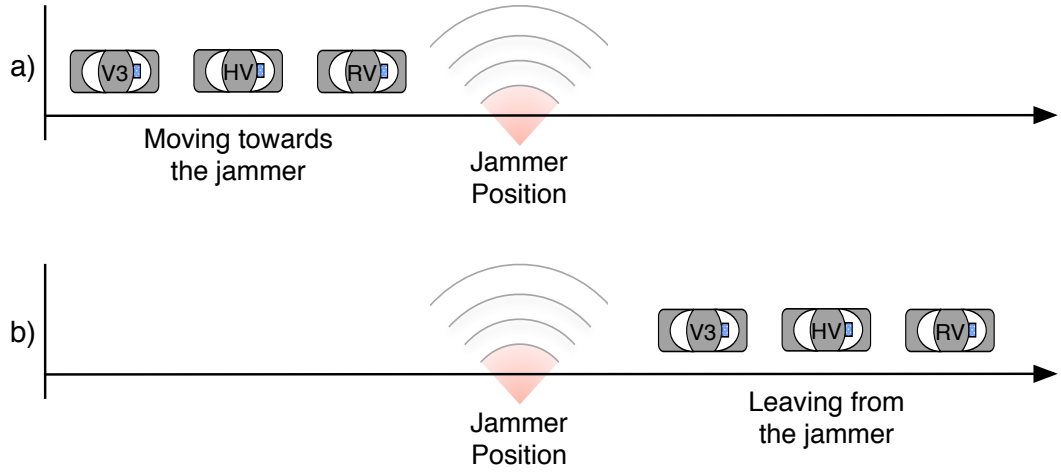


FIGURE 3.1: Field experiment scenario [39]

Figure 3.2 shows the data that was logged by V3 of BSMs sent from the RV and HV just before the medium was completely jammed. When not affected by jamming, BSMs from the RV and HV were received by V3 with the expected time spacing. Once the vehicles entered the jamming area, gaps in reception were observed, as expected. However, unexpected bursts of BSMs were logged following small gaps of reception. After careful investigation of the timing and content of packets, we could confirm that the bursts were due to OBU message queuing as the medium was jammed, i.e., BSMs were queued during jamming. After jamming the transmission queues flushed with bursts of BSMs. These bursts did not follow the 10BSM/s

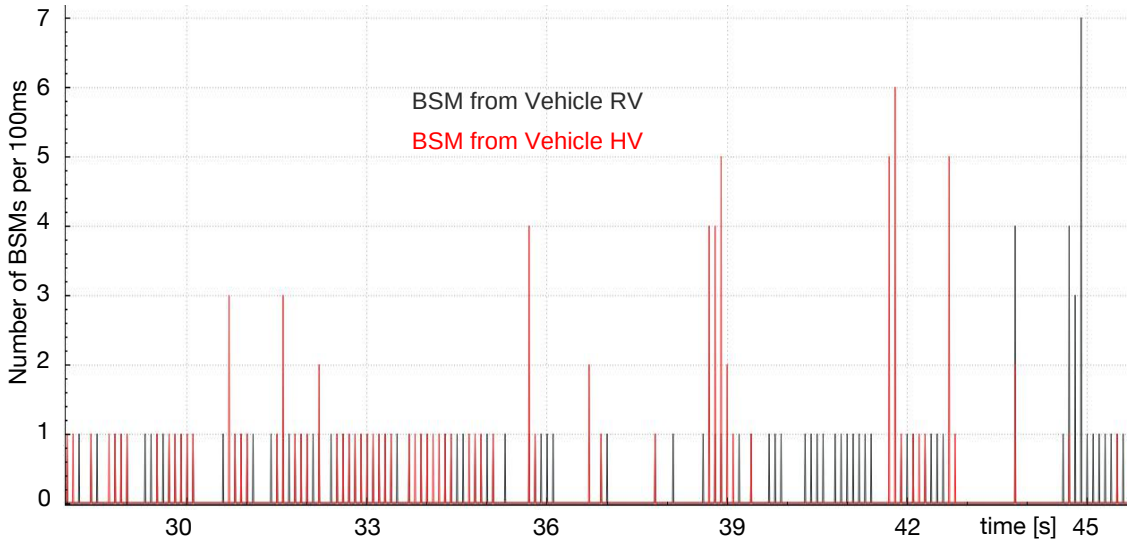


FIGURE 3.2: Field test observations logged by vehicle V3

rate. This queuing behavior and subsequent burst could be exploited by the hybrid jammer described next.

3.3 A NEW HYBRID JAMMER

The observations described in the previous section inspired a new jammer, that combines properties of constant, deceptive, and random jammers. The jammer sends continuous random bits like a constant jammer, however, they appear as regular packets, without following the channel access protocol, like a deceptive jammer. In addition the jammer is dormant for most of time and only jam for specific durations, e.g., half a second to a few seconds. This makes it appear like a random jammer, however, it will be shown later that the time and duration of jamming is carefully selected. During jamming all nodes believe that their inability to access the medium is because other nodes are transmitting their packets. Therefore, legitimate nodes will not be able to transmit any packets and queue them instead, until jamming stops and the medium becomes available again. No packets will be lost as long as the queues of the nodes do not overflow. Due to this overall behavior, no messages of legitimate nodes are destroyed and the jamming cannot be detected as a malicious attack by mechanisms relying on packet error rates or delivery ratios.

3.3.1 *Hybrid Jammer Properties*

The behavior of the hybrid jammer can be formally described as follows: Assume the size of transmission queue of the OBUs is q. Let ΔT_{jam} denote the jamming duration, and t^s_{jam} and t^e_{jam} the time at which jamming starts and ends respectively. Thus, the time interval ΔT_{jam} can be defined as $\Delta T_{jam} = t^e_{jam} - t^s_{jam}$. Theoretically speaking, jamming for ΔT_{jam} will result in each OBU that is affected by jammer to queue $m = \Delta T_{jam}/0.1$ BSMs, where $0.1s$ is the BSM time spacing. An attacker can take advantage of jamming-induced queuing by:

1. Minimizing being detected by selectively choosing ΔT_{jam} to intentionally cause BSM delays suiting its attack objectives, e.g., selecting the smallest BSM delay that renders safety applications useless. In the context of timeliness of BSM messages, a delay in reception d_i will occur for each BSM_i queued due to jamming for a duration of ΔT_{jam}. The minimum delay of a queued BSM_i is computed by

$$d_i \geq \Delta T_{jam} + \sum_{1}^{i} t_{min}, \quad 1 \leq i \leq m \tag{3.1}$$

   where t_{min} is the lower bound on the BSM transmission time from a specific OBU.

2. Ensuring no BSMs of legitimate vehicles are lost during the jamming period. This can be achieved if the duration is short enough to not overflow an OBU's transmission queue, i.e., if $m \leq q$, and if no congestion occurs due to affected OBUs attempting to flush their queues.

3. Making innocent vehicles appear to be misbehaving. Once jamming stops OBUs send all queued BSMs in a burst that does not follow the 10BSM/s rate, flushing in this way makes them appear to be selfishly misbehaving on the medium.

The hybrid jammer's function can be explained using the timing model of EEBL, shown in Figure 2.9. The attacker can cause EEBL to fail it delays more than x BSMs so that they are not received by the safety application in proper time. Furthermore,

if the number of queued BSMs x is less than or equal to the queue size q, then any PDR-based approache will fail to detect the jammer, as no BSMs of legitimate nodes will be lost. Finally, the queuing vehicles will appear as misbehaving due to their OBUs flushing their queues after jamming stops.

3.3.2 *Jamming Impact on Queuing*

As indicated in the discussion of Figure 3.2 queuing of BSMs due to jamming was observed in real tests using commercially available OBUs. To validate the hybrid jammer's effect on queuing and to investigate if queuing is deterministic, we conducted experiments using three Locomate Classic OBUs from Arada Systems [20]. One OBU was programmed to act as the hybrid jammer, the second served as the RV and the third as the HV. The experiments were conducted in a controlled environment with no objects interfering with communications. The test parameters used are shown in Table 3.1.

TABLE 3.1: Hybrid Jammer Parameters

OBU Model	Arada Systems LocoMate Classic
Number of OBUs	3 (2 OBUs for two vehicles and 1 for the stationary jammer)
BSM generation	10 packets/s
Channel	Safety Channel 172
Transmitter power	21 dBm
Data rate	6Mbps
Jammer power and data rate	18 dBm, 6Mbps

The impact of jamming with $\Delta T_{jam} = 1, 2, 3$ and $4s$ of a typical experiment can be seen in Figure 3.3, where the number of BSMs that the HV received from the RV per 100ms is shown. After each jamming period a burst of BSMs, consistent with the number of BSMs expected to have been queued based on ΔT_{jam}, can be observed. However, after careful examination the experiment also revealed that jamming in practice is not as precise as in theory, and we observed several factors that introduced variability.

First of all, the time from starting the jammer until it effectively jammed the medium was nondeterministic, as it was not possible to start jamming precisely at

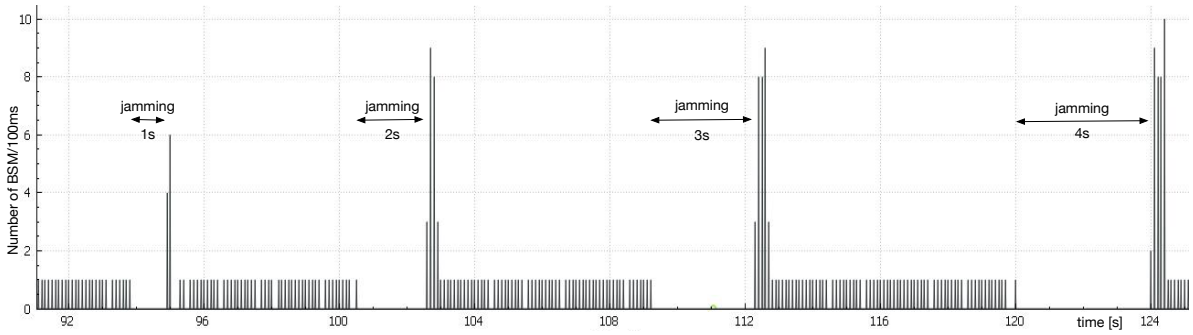


FIGURE 3.3: Queuing effect for jamming periods of $1, 2, 3$, and $4s$

the time intended. We attribute the observed differences to process initialization delays, the runtime overhead of the Arada LocoMate Classic's operating system, which is Linux based, and the overhead associated with the jammer program. This delay can be observed in the scenario with $\Delta T_{jam} = 1$, which actually resulted in an effective jamming period slightly longer than 1s.

The second factor that created nondeterminism was attributed to the Arada Locomate Classic OBUs way of flushing their buffer. Specifically, experiments revealed that BSM spacing during flushing was on average 12.5ms, with minimum and maximum observed spacings as 10ms and 16ms respectively, and standard deviation of 1.6. In Figure 3.3 this behavior was responsible for several spikes after the jamming period rather than one large spike.

Thirdly, it should be noted that the figure only shows BSMs sent by the RV and received by the HV. The HV also queued messaged during jamming, which also accessed the medium using CSMA/CA. However, due to the low utilization of the medium this should have had minimal impact on the data in the figure. The experiments further suggested that the Arada Locomate Classic OBUs queued up to 40 BSMs before messages were dropped.

3.4 DESIGN CONCEPTS

From this research point of view, all jamming models presented in Section 2.5.2 can be classified into two different models: destructive and non-destructive jamming models. Destructive jammers interfere with communication between nodes, thus

they may destroy messages sent by legitimate nodes during the jamming period. A non-destructive jammer blocks legitimate nodes from accessing the medium, thereby forcing these nodes to queue messages in their transmission queues. The hybrid jammer is the main jamming model for this research. It is an example of a truly non-destructive jammer, as it impacts reception of BSMs by delaying, but not destroying them. This delay in reception could have sever impact on safety application.

3.4.1 *Detection of Hybrid Jammer*

Detecting the hybrid jammer is based on two metrics. The first is the difference between the time stamps of the creation of the most recently saved BSM and the currently received BSM. Such time stamp is included in a BSM field called *Dsecond*. If this difference is significantly less than 100ms, this vehicle is considers to be misbehaving. The second metric is the difference between the time stamp of the HV and that in the recently received BSM. This difference allows identifying the number of missing BSMs. If this difference divided by the BSM period of $100ms$ is approximately equal to the number of BSM omissions identified, then this vehicle is a victim. The Hybrid Jammer Detection Algorithm (HJDA) is explained in detail in [61].

3.4.2 *Attack Model*

A scenario involving vehicles reacting to a hazard in the absence of an attack is shown in Figure 3.4. Vehicles RV1 and RV2 observing a hazard react, causing their BSMs to indicate a braking event. Vehicle HV is assumed to not have visual contact and thus its safety application relies on messages from the RVs, which both consistently indicate the event.

Next, consider the scenario depicted in Figure 3.5, where a hybrid jammer is positioned on the roadside next to RV1 and RV2. This jammer, A1, jams for a period of ΔT_{jam} in coordination with the creation of the hazard in front of the RVs. At the same time a collaborating attacker vehicle, A2, starts sending false BSMs indicating no event to the HV. In the absence of jamming, the false values of A2 would be

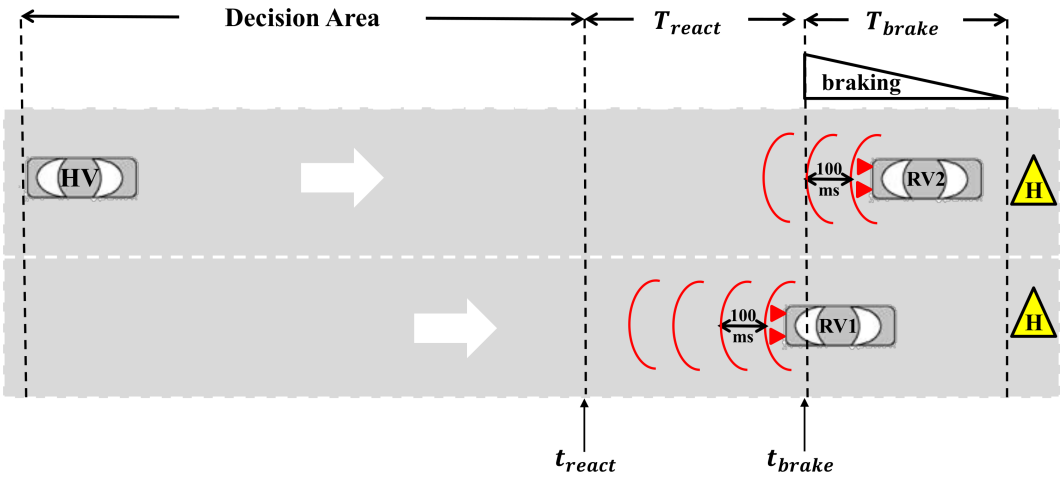


FIGURE 3.4: No attack

outvoted by the correct values of RV1 and RV2 in the voting algorithm executing in the HV.

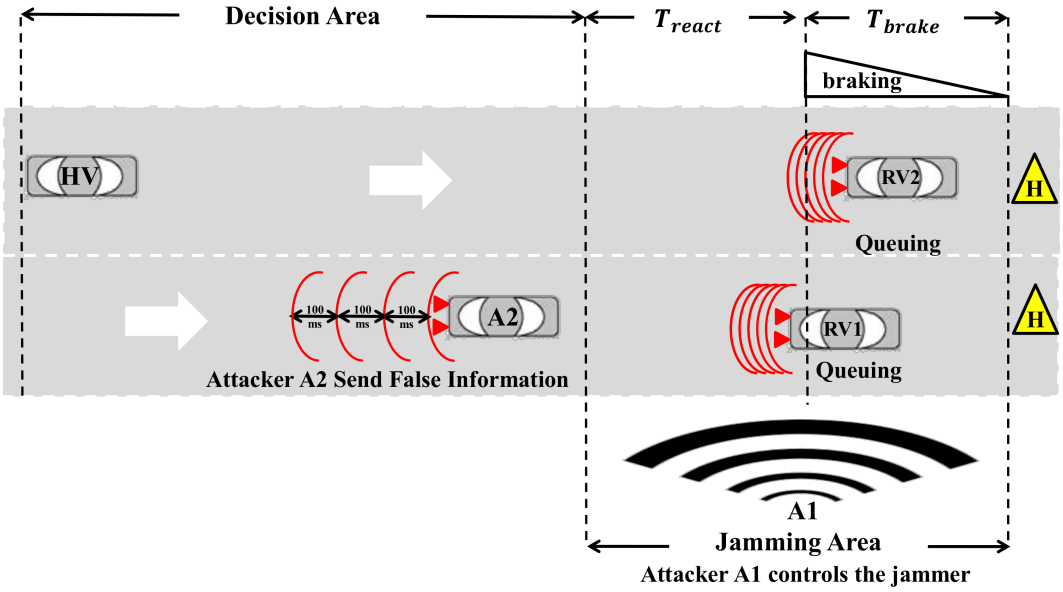


FIGURE 3.5: Attack causing message queuing

In the presence of jamming the above behavior changes. Specifically, the hybrid jammer will force the RVs to queue their BSMs during jamming period ΔT_{jam}. The attacker A2, positioned outside of the jamming area, will be able to stack the voting set of the HV with false values. The HV will have to make a decision to notify the

driver of the event before it is too late to react. This time is approximated by

$$t_{safety} = t_{now} + T_{safety} \tag{3.2}$$

where t_{now} is the current time and

$$T_{safety} = \frac{loc_{HV} - origloc_{RV1}}{speed_{HV}} - T_{react} \tag{3.3}$$

where $speed_{HV}$ is the current speed of the HV, and loc_{HV} and $origloc_{RV1}$ are the current and original locations of the HV and RV1 respectively. We can define the EEBL application reliability as the probability of the algorithm taking a correct decision at or before t_{safety}.

3.4.3 *Enhanced Voting Algorithm (EVA)*

The EVA is built on an architecture consisting of multiple components. A core component is the *dispatcher*, which is similar to the dispatcher used in the system model of [38]. It starts a separate thread for each distinct event e_j observed, i.e., upon the first occurrence of e_j reported in a BSM by some vehicle. The dispatcher forwards BSMs to the threads corresponding to events e_j if the RV sending the BSM is located in the event detection zone. In [38] this zone is determined by a so-called filter, which uses metrics such as distance from, and lane of an event.

The EVA, which is running in the thread associated with each e_j, is shown in Figure 3.6. The algorithm consists of two stages: an investigation and a voting stage.

Investigation: This stage deals with hybrid jamming attack and/or misbehavior detection. The EVA calls the HJDA described in Subsection 3.4.1, which identifies if an RV is a victim of a hybrid jammer attack or a misbehaving node. For each vehicle the HJDA saves the last BSM received and keeps track of the number of BSM omissions. This can be done using watchdog timers, since a BSM is expected from each vehicle approximately every 100ms. The HJDA controls Misbehaving Vehicle flags, MV_i, one entry for each surrounding RV. The MV_i flags are initially cleared when EVA is invoked and are used to differentiate between misbehaving and

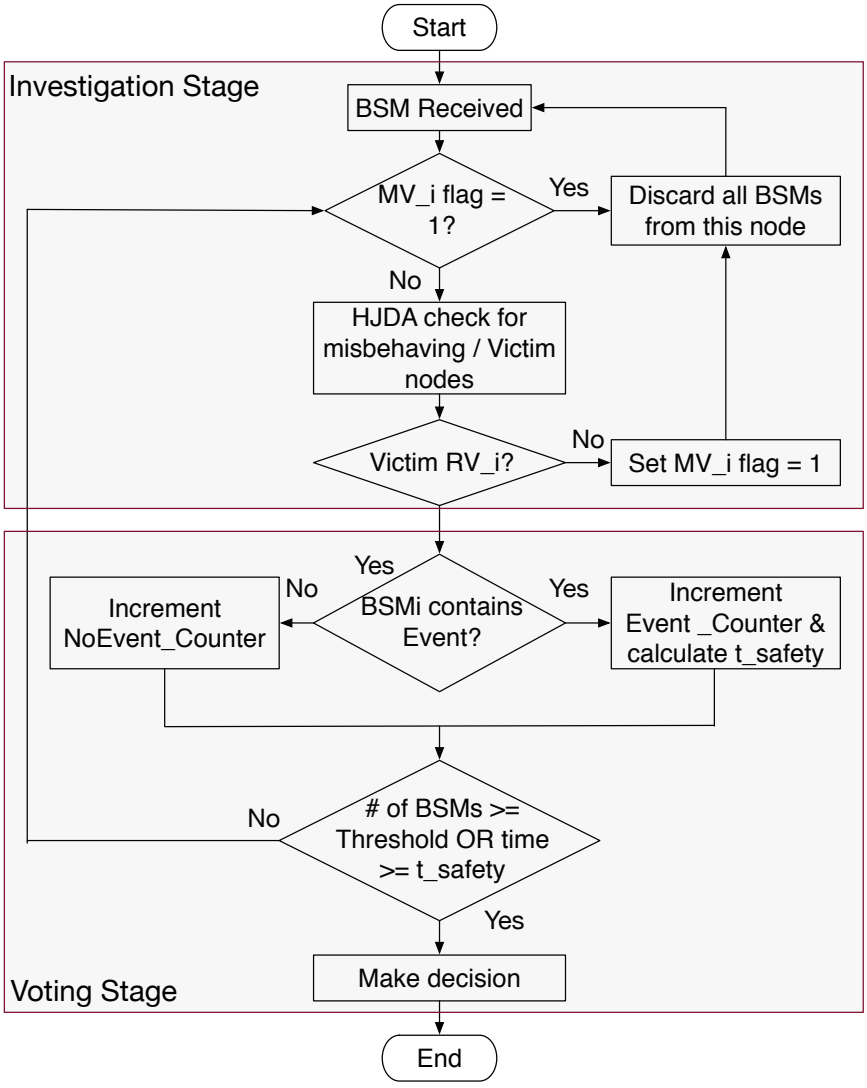


FIGURE 3.6: Enhanced Voting Algorithm for Event e_j

legitimate RVs. When a BSM is received and its corresponding $MV_i = 1$, indicating misbehaving, the BSM is discarded. Otherwise, the HJDA is called to check whether RV_i is misbehaving or if it is a victim of hybrid jamming.

Voting: If RV_i passes the investigation stage, its value is added to the voting set. It should be noted that, due to the timing checks in the HJDA of the investigation stage, this not only includes BSMs with regular 100ms transmission rates, but also those from victim RVs flushing their messages queued during jamming. These latter BSMs will be discarded if they are considered to be outdated.

Recall that t_{safety} is the critical time by which the HV needs to take action. The value for t_{safety} is updated for each BSM indicating the event. A final decision using voting is taken if 1) t_{safety} is reached or 2) if the threshold on the cardinality of the voting set is reached. Due to the investigation stage, the EVA bases its voting decisions only on values coming from non-misbehaving and victim vehicles.

3.5 PERFORMANCE ANALYSIS

In this section we will investigate the performance of previous voting algorithms subjected to the hybrid jammer. Then the performance enhancements of the EVA will be presented. But first the field test assumptions are introduced.

3.5.1 *Setup of Experiments*

Whereas the theoretical impact of the hybrid jammer on the vehicles is clear, as it is defined directly by the attack model and its associated queuing behavior described in Section 3.4.2, the impact of the jammer in the field needed to be investigated. Therefore, in order to validate the EVA, field experiments were conducted and the results were collected and analyzed. The experimental setup consisted of the scenario compatible with that shown in Figure 3.5. Specifically, vehicles representing one HV, two RVs, and attacker A2, were equipped with LocoMate Classic OBUs from Arada Systems [20]. All OBUs of vehicles used the standard transmission rate of 10 BSMs per second and a transmission power of 23 dBm using Safety Channel CH172. The RVs were configured to send BSMs announcing that an event occurred. On the other hand, the attacker A2, located in the detection zone, sent BSMs falsely indicating no event. The OBU in the HV executed the EVA. An additional OBU was configured to act as a hybrid jammer capable of operating with different transmission powers, data rates and jamming periods.

The experiments were conducted in a controlled configuration, where the vehicles were positioned as in Figure 3.5, however they were stationary. This configuration mimics the worst case real scenario, in which the jammer can control the RVs precisely, without affecting attacker A2. The jammer was placed directly

next to the RVs and produced 1 dBm of jamming power for a duration of 1.5, 2 and 4.5 seconds. The attacker A2 and the HV were placed outside of the jamming area, 80 meters from the RVs. The reason for conducting the experiment stationary rather than on moving vehicles was multifold. During many hours of field testing it proved to be very difficult to maintain constant speeds, and thus distances between all vehicles. Furthermore, we needed to shield the experiment from the impact of the road geometry, such as curves and elevation changes, as well as that of unrelated road traffic, in the absence of a dedicated test site. A summary of the experiment parameters is given in Table 3.2.

TABLE 3.2: Field Experiment Configuration Parameters

OBU Model	Arada Systems LocoMate Classic
Number of OBUs	5 (4 OBUs in four vehicles, 1 OBU as stationary jammer)
Test range	straight one-lane road
Jammer position	2m from the RVs
Distance: HV to RV1	80m
Vehicles speed	0 m/s (Fixed)
BSM generation	10 BSM/s
Channel	Safety Channel 172
Transmitter power	23 dBm
Data rate	3Mbps
Jammer power & data rate	1 dBm, 3 Mbps

3.5.2 *Previous Voting Approaches*

Figures 3.7 and 3.8 show several graphs, each of which represents certain contributions to the voting set with respect to different jamming periods.

In the absence of attacks, the messages contributed to the voting set by RV1 and RV2 are shown in graph *Normal* 2-*RVs* in the figures. With no message losses this graph would be linear. However, during the specific field test represented in the graph, three BSMs were lost.

Next we consider the impact of attacks on the voting set using the attack scenario shown in Figure 3.5. Note that attacker A2 is not affected by jamming, whereas RV1

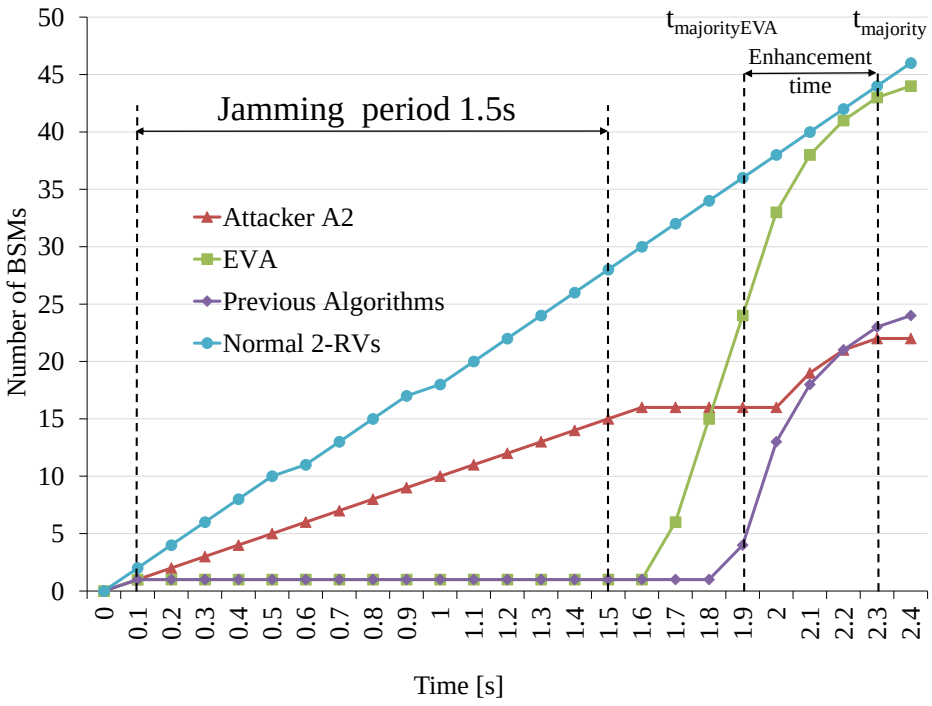


FIGURE 3.7: Performance: 1.5 second jamming with Arada OBU

and RV2 are in the jamming zone. The measured contribution of attacker A2, who is sending false data, is shown in graph *Attacker A2*.

During different jamming periods no BSMs from RV1 and RV2 were received by the HV and only BSMs from A2 are visible in the voting set. Recall that the voting algorithm is running on the HV. However, due to the omission of BSMs from RV1 and RV2 as the result of message queuing, the threshold of the voting set of previous algorithms, e.g., [38, 39], cannot be reached yet to make a decision.

Once the jamming period ends, BSMs queued in the RVs are flushed, as described in Subsection 3.3. Incidentally, this flushing is responsible for the flat part of graph Attacker A2, as A2 could not access to the medium. Recall that with a BSM transmission rate of 10Hz and jamming periods of 1.5 and 2 seconds, approximately 15 and 20 BSMs were queued by each RV respectively. According to [45] the time-to-live of a BSM should be no more than 500ms. Older messages are considered to be outdated. This implies that the voting algorithm running on the HV will consider only the most recent 5 BSMs of the queued BSMs in each RV. The contributions of RV1 and

RV2 to the voting set of algorithms using this real-time constraint is depicted in graph *Previous Algorithms* in both figures.

As stated before, a voting decision needs to be made either once the voting set cardinality threshold is reached, or no later than time t_{safety}. A voting algorithm can make a correct decision only once the voting set contains a majority of correct BSMs. This point is reached in Figure 3.7 and 3.8 at time $t_{majority}$. If $t_{safety} < t_{majority}$ the voting algorithm will come to the wrong decision, otherwise a correct decision is made.

3.5.3 *Performance Evaluation of the EVA*

One of the key issues of voting is the fact that messages may arrive in bursts due to jamming and that outdated messages could be discarded (as argued in [45]). Recall that if algorithms consider time, then RVs will be considered as misbehaving nodes during bursts. Consequently their BSMs would be discarded, even if they indicate an event. If time is not considered, then the voting set could be highly affected by misbehaving nodes, as they would disproportionally stack the set. The EVA has the capability to resolve these conflicts.

The graph *EVA* in Figures 3.7 and 3.8 shows the performance of the new algorithm. During the jamming periods no BSMs from the RVs are added to the voting set. However, once jamming stops, the bursts of queued BSMs from RVs are added to the voting set, as they arrive. The EVA can make a correct decision once majority is reached at time $t_{majorityEVA}$. The figures show the difference in time the EVA and the previous algorithms reach their corresponding $t_{majority}$, indicated by *Enhancement time*. As can be seen, the EVA outperformed other voting algorithms by 0.4s and 0.9s for the 1.5s and 2s jamming periods respectively. For safety applications such improvement could have significant impact, for example the 0.9s enhancement is approximately equal to typical reaction times. From a safety application reliability point of view a reliable decision can be made by the EVA at time $t_{majorityEVA}$, whereas the previous algorithms will make a wrong decision during the half-closed time interval $[t_{majorityEVA}, t_{majority})$.

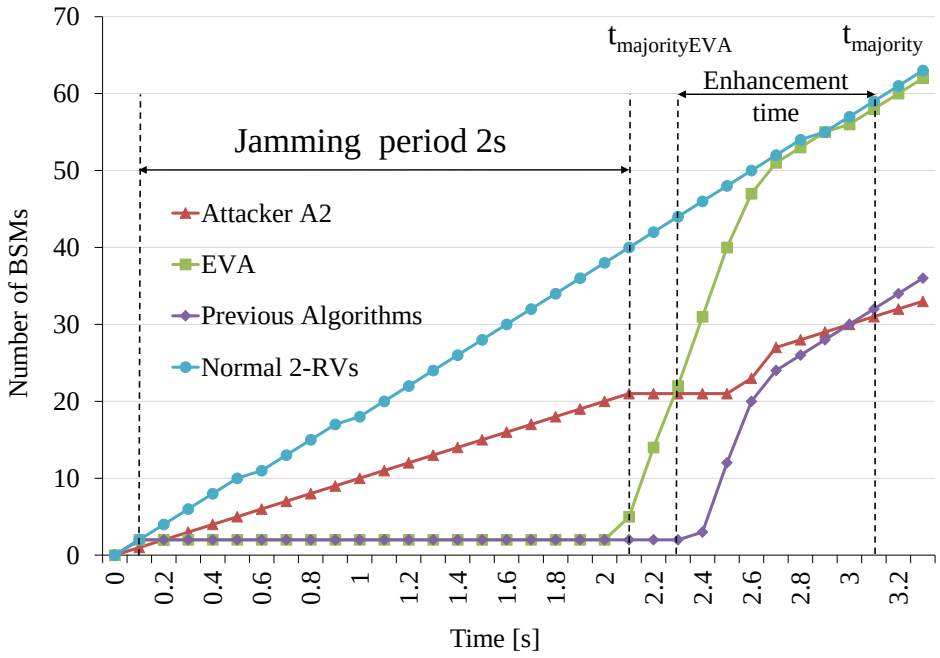


FIGURE 3.8: Performance: 2 second jamming with Arada OBU

The performance of the EVA in Figures 3.7 and 3.8 was derived with data captured from Arada LocoMate OBUs, which were evaluated to have a queue size of 40. An attacker knowing the OBU queue size could force a worst case behavior by causing the queue to overflow. Such scenario is shown in Figure 3.9. Here the jamming period of 4.5 seconds is sufficiently long to queue 40 messages and drop 5. Recall that during 4 seconds 40 BSMs are sent. The previous algorithms will discard all queued messages. Together with the 5 messages dropped due to the *newest packet dropped* behavior [15], it will have no messages to consider. The EVA on the other hand will consider all queued messages, leading to the largest enhancement time of 3.3 seconds, as can be seen in the figure.

Running the EVA, which implies also the executions of the HJDA algorithm, imposes computation overhead on the OBUs. This overhead is constant and negligible for each BSM in HJDA and the EVA, with respect to updating data structures. The delay associated with achieving the threshold in Figure 3.6 is dependent on messages received from vehicles in the detection zone. However, this time is bound by the T_{safety}, as described in Section 3.4.2.

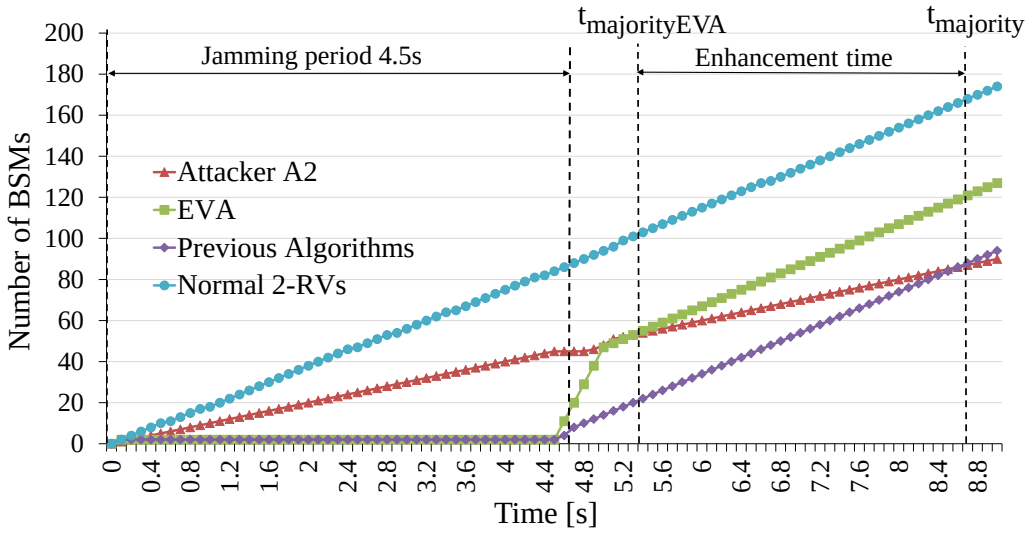


FIGURE 3.9: Performance: 4.5 second jamming with Arada OBU

3.6 CONCLUSIONS

This chapter presented a new Enhanced Voting-based Algorithm to improve reliability of DSRC safety applications operating in hostile environments. A hybrid jammer, potentially causing safety applications to fail, was considered and its detection algorithm was discussed. This jammer type is not only capable of forcing nodes to queue messages, but also making these legitimate nodes appear as misbehaving. Field experiments based on an attack model were conducted to demonstrate the impact of the hybrid jammer on forced queuing in specific commercially available OBUs. The EEBL safety application was used as an example during experiments. The results observed showed that the EVA was capable of significantly reducing the application's decision times, thereby improving application reliability. In worst case scenarios, which an intelligent attacker could provoke, the improvements of the EVA were significant. During experiments enhancements of up to 3.3 seconds were observed. In the context of safety critical applications, such improvements could have significant impact on avoiding accidents and saving lives.

CHAPTER 4

Sybil Attack Detection Algorithms

In this chapter the Sybil attack, which is considered to be one of the most severe attacks in VANET, is investigated. In this attack a malicious node forges many fake identities to fool safety applications. Two algorithms are presented. First, a passive Sybil detection algorithm is proposed that improves the resilience against Sybil attacks in a static power environment. The simulations shows that passive detection algorithms might be ineffective in dynamic power environment. Second, an active Sybil attack detection algorithm is presented that can locate Sybil nodes without adding special hardware or information exchanges. Unlike previous detection approaches, the algorithm is capable of Sybil detection even in dynamic power environments. The active algorithm was evaluated in the field using vehicles equipped with Arada LocoMate Classic OBUs.

4.1 RELATED RESEARCH

The literature on detecting Sybil attacks in VANET shows a variety of approaches. These approaches can be classified as follows.

4.1.1 *Resources Testing*

The objective of resource testing is to determine if a number of nodes, e.g., Sybil nodes, have less resources than would be expected if they were legitimate independent nodes. This method falls into three categories. The first category is *Radio Resource Testing*, presented in [35]. In this method, the node that wants to detect the Sybil nodes assigns a channel to each neighboring node to broadcast messages on it and randomly chooses a channel to listen. If it is a legitimate node, it should receive the message and response on the same channel. Otherwise, the malicious node can not send a response message for its Sybil nodes simultaneously on different channels. The second category is *Computational Resource Testing*. When a node wants to detect

Sybil nodes it sends a puzzle to be solved to all nodes. As the malicious node and its Sybil nodes sharing resources, nodes failing to solve a puzzle are marked as Sybil nodes [36]. Finally, *Identification Resource Testing* is discussed in [37]. Here a node can detect Sybil attacks by saving the MAC addresses of neighbor nodes in a list. If a node is detected with a MAC address not recorded in this list, it is identified as a Sybil node. None of these methods are applicable in VANET.

4.1.2 *Ranging Methods*

Ranging methods are proposed to calculate the distance between a transmitter and a receiver, and can be classified into two categories:

Received Signal Strength Indicator (RSSI) — Sybil attacks can be detected using the Received Signal Strength Indicator (RSSI) propagation model as described in [46] [47]. In this method, the receiving node uses the received signal strength as the basis for calculating the distance of the sending node. If the calculated distance differs from the distance implied by the two nodes' GPS coordinates, the sending node may be a Sybil node. The authors in [48] present an approach that is composed of two complementary techniques assuming all vehicles use the same transmission power. They first use RSSI to estimate the distance between two nodes using the Friis model. When incoherent signal strengths are observed, a second technique using a "distinguishability degree metric" is used, which is based on observing differences of two nodes over time. However, any intelligent attacker who can manipulate the GPS coordinates and power levels to appear consistent, will be able to fool these approaches.

Time Based Methods — Time-based methods such as Time Of Arrival (TOA) and Time Difference Of Arrival (TDOA) are presented in [49] [50]. Here the estimated distance between two nodes is based on the signal propagation time. However, this requires an accurate real-time clock synchronization between the transmitter and the receiver, which may not be a realistic assumption [51].

4.1.3 *Neighbor Information and Collaboration*

The authors in [52] proposed a technique in which each node exchanges group information of its neighbors periodically with other nodes. Each node performs the intersection of these groups. If nodes observe very similar neighbors over a longer time, they flag these matching neighbors as Sybil nodes. The assumption is that it is unlikely that two nodes have the same set of neighbors for a time surpassing a certain threshold. However, this approach has limited detection capabilities and adds more communication/message overhead to the system.

An intrusion detection system approach to rogue node detection was introduced in [53]. Their anomaly-based detection uses node driving information to calculate metric such as average flow, density and location, which is then used as a base line shared with other nodes. This collaboration allows comparisons of traffic flow averages. Flow averages that appear extreme are rejected and reported. However, Sybil nodes, especially if there are many, could affect these computations. Furthermore, these nodes can behave normal, yet inject and broadcast false data, e.g., a brake status event indicating hard braking.

4.1.4 *Road-Side Unit (RSU)*

In [54] the authors introduced a so-called "Robust method of Sybil Attack Detection (RobSAD)" in urban VANETs. Because their Sybil nodes have the same location and direction all the time, their group behavior is assumed suspicious. The authors suppose that authorized RSUs are distributed over part of this area. These RSUs broadcast digital signatures with timestamps to vehicles in their range. Honest nodes have independent trajectories and collect the signatures received from authorized RSUs. Sybil detection is achieved by analysis of the neighboring nodes' signatures.

A timestamp-based approach using RSU support to detect a Sybil attack is presented in [55]. The authors assume that it would be unlikely for two vehicles to pass by two or more different non-proximate RSUs at similar times. Sybil attack detection is triggered when multiple messages from different vehicles (Sybil nodes)

contain similar series of timestamps. However, if RSUs are located at intersections or in the absence of RSUs, it may make Sybil attack detection difficult or impossible.

4.1.5 *Cryptography and Authentication*

A mechanism using public key cryptography and authentication to prevent Sybil attacks is described in [56] [57]. Specifically, asymmetric cryptography is used. Signatures are combined with digital certificates, issued by a Certification Authority (CA), with one CA for each region. The CAs communicate through secure channels and keep track of issued certificates used for signed messages. Only messages with valid certificates are considered and invalid messages are ignored. However, this mechanism requires that each node is assigned one certificate at a time. On the other hand, these certificates should be changed frequently for privacy. It is unrealistic and difficult in VANETs to deploy Public Key Infrastructure as there is no guarantee that the appropriate infrastructure will be present and the approach is time consuming.

In [58] a scheme is proposed that uses encryption and four security aspects. 1) Authentication - before any message is transmitted, a vehicle should receive its public authentication key. 2) Non-repudiation - the vehicle uses a group authentication key and an encryption function, which it then sends along with the original message to other vehicles and RSUs. 3) Privacy - it is not mandatory for each member to have the private information of other members. 4) Data Integrity - receiving nodes verify the authenticity of members using the signature. The major drawback of this approach is that most operations are done in the CA and do not run at the node itself, which may not be practical in all situations. Furthermore, it is not possible to discover the location of malicious nodes.

4.2 ASSUMPTIONS AND ATTACK MODEL

In this research the following assumptions will be made:

1) An attacker can have more computational power and flexibility than ordinary OBUs or RSUs. It may tune its transmission power to achieve certain signal strengths

at a target vehicle's receiver. This assumption can be easily justified by the availability of devices satisfying such properties.

2) An attacker may inject false information or other fields into a BSM. This includes manipulation of GPS coordinates. Thus, there are no restrictions imposed on the attacker's conduct. We have experimented extensively with such manipulations using *Arada LocoMate Classic* [20] OBUs.

3) Attackers can use more than one certificate to send messages. The justification for this assumption is the possibility that any attacker could possess portable DSRC devices, such as the *Arada LocoMate Me* [20], or use stolen devices. Furthermore, sending a BSM does not require authentication if the goal is to reduce overhead, as may be the case in high traffic density situations.

4) Similar to the research presented in [48], we assume that honest vehicles are equipped with standard OBUs, where the antenna's properties and gains are fixed and known.

5) The assumptions about the transmission power of the vehicles in the VANET, which may be either static or dynamic, directly affect the attack model. Therefore, investigations on Sybil detection under both static or dynamic power environment assumptions have been made.

Figure 4.1 depicts an attack scenario with an HV, two honest RVs, and a malicious vehicle projecting four Sybil nodes. Whereas RV1 and RV2 send BSMs every 100ms,

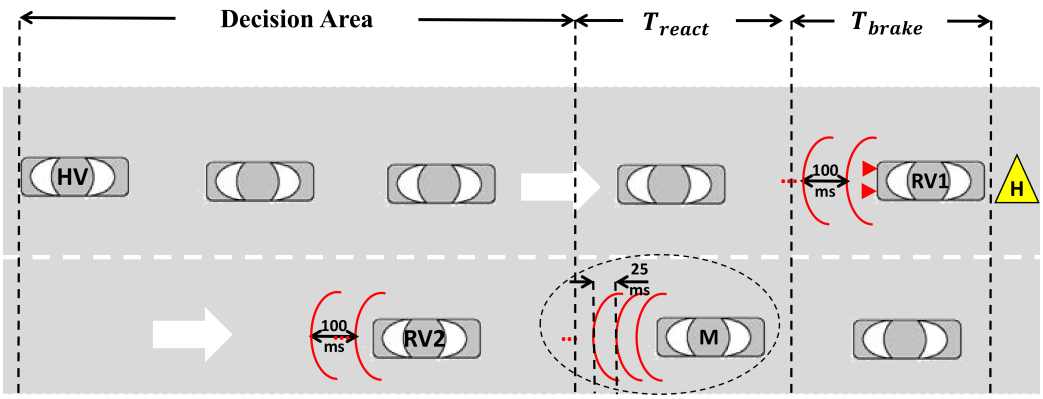


FIGURE 4.1: Sample attack scenario

the malicious node sends one BSM each 25ms, alternately claiming to be another (non-existing) vehicle. Now assume that RV1 brakes hard due to an observed hazard.

It consequently sends this "hard braking" event in its BSMs. Assume the HV runs a safety application using voting, such as presented in [39] or [40]. The goal of these voting algorithms is to collect BSMs from vehicles in the vicinity of the reported event to see if they also reacted. If a certain threshold of vehicles report the braking event, the HV assumes it is legitimate. However, the malicious node, with its Sybil nodes, can inject BSMs contradicting the event, thereby affecting the vote of the HV.

4.3 PASSIVE SYBIL DETECTION

In this section a passive Sybil detection algorithm is presented under the assumption of a static power environment. The term *"passive"* implies that honest nodes will not use any active mechanisms like extra probe messages, and will only rely on the standard BSMs.

4.3.1 *Notation and Basic Concept Definitions*

Let $\mathbf{R}_i$ be the set of remote vehicles RV_j from which BSMs were received by HV_i. Initially this neighborhood set $\mathbf{R}_i$ is empty. Furthermore let S denotes the receive sensitivity (also called Rx sensitivity) of the HV, which was previously defined as the minimum input signal required at the receiver's antenna to produce a minimum required signal-to-noise ratio at the output of the receiver. A BSM is only received by the HV if the BSM's received signal strength is not less than S. Of interest is the HV's maximum reception distance for a given S, which is denoted by $d_{max}(S)$. In free space, this distance defines the radius of a circle representing the reception area. Thus, the HV can receive BSMs from any node within this circle. The signals from vehicles outside of the circle will be too weak to be received. The distance $d_{max}(S)$ can be computed by

$$d_{max}(S) = \sqrt{G \times P_{snd}/S} \tag{4.1}$$

where P_{snd} is the standard transmission power and G is the gain, as computed by [59]

$$G = G_{snd} \times G_{rcv} \times \lambda^2/(16\pi^2) \tag{4.2}$$

Here λ is the wavelength and G_{snd} and G_{rcv} are the sender and receiver gains, which are both known as indicated in Subsection 4.2 Assumption 4.

Assume a specific host vehicle HV_i with Rx sensitivity S_i and $\mathbf{R}_i$, the set of remote vehicles RV_j from which BSMs were received. The number of actual vehicles from which BSMs were received, denoted by $N_{act}(S_i)$, is simply the cardinality of $\mathbf{R}_i$, i.e.,

$$N_{act}(S_i) = |\mathbf{R}_i| \tag{4.3}$$

The expected number of RVs is calculated using the GPS coordinates of each RV_j that lies within $d_{max}(S_i)$ of the HV. Thus

$$N_{exp}(S_i) = \sum_{RV_j \in \mathbf{R}_i} f_j \tag{4.4}$$

where

$$f_j = \begin{cases} 1 & \text{if } d_j \leq d_{max}(S_i) \\ 0, & \text{otherwise} \end{cases} \tag{4.5}$$

and d_j is the distance between HV_i and RV_j, based on GPS coordinates. Let m be the difference in actual and expected numbers of RVs. Therefore, $m = |N_{act}(S_i) - N_{exp}(S_i)|$.

4.3.2 *Passive Sybil Detection Algorithm*

The passive Sybil detection algorithm, which executes on the HV, offers Sybil detection based on changing Rx sensitivity of the HV, and can identify both the malicious and Sybil nodes. As stated before, this detection works under the assumption of a fixed power environment, however, this assumption applies only to honest nodes, and no restrictions are made about the power of malicious nodes.

The Sybil detection algorithm shown in Figure 4.2 has two parts. The first part of the figure is identified by the shaded box. Here BSMs are received and the IDs of their sender's RV_j are added to $\mathbf{R}_i$, unless they are already in the neighborhood set. The time over which set $\mathbf{R}_i$ will be defined is thus approximately 100ms, the time between two BSMs from the same vehicle.

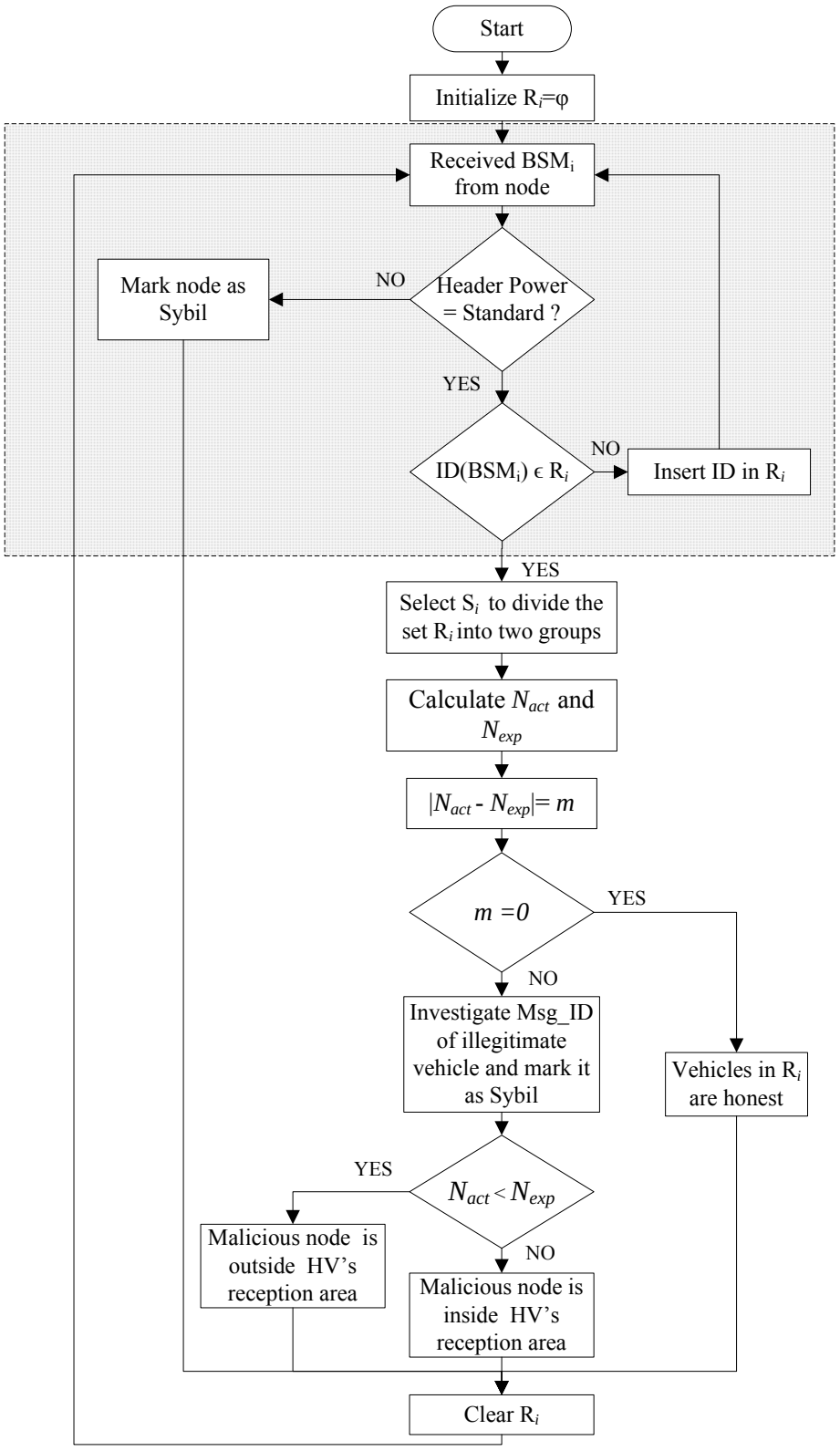


FIGURE 4.2: Passive Sybil detection algorithm

In the first part, upon receiving a BSM, the algorithm checks the power field in the BSM header [10] to see if it deviates from the standard power value of the fixed power environment. If it does, the node is marked as a *Sybil* node. The algorithm continues to receive BSMs until it receives one from an RV whose ID is already in $\mathbf{R}_i$. This reception of a second BSM from the same RV marks an interval of approximately 100ms.

In the next part, the algorithm calculates the HV sensitivity S_i which is able to divide the RVs $\in \mathbf{R}_i$ into two groups. The first group includes all RVs within the HV's reception area, and the second group contains all other RVs. Figure 4.3 depicts an illustrative scenario. Assume that the HV that executes the proposed detection algorithm has a receiver sensitivity S_1. In this case the set $\mathbf{R}_i$ includes all RVs located within $d_{max}(S_1)$ as shown in Figure 4.3 a). Whereas when the algorithm selects a sensitivity S_2, these RVs in $\mathbf{R}_i$ will be divided into two group based on their distance from the HV compared to $d_{max}(S_2)$. This case can be seen in Figure 4.3 b).

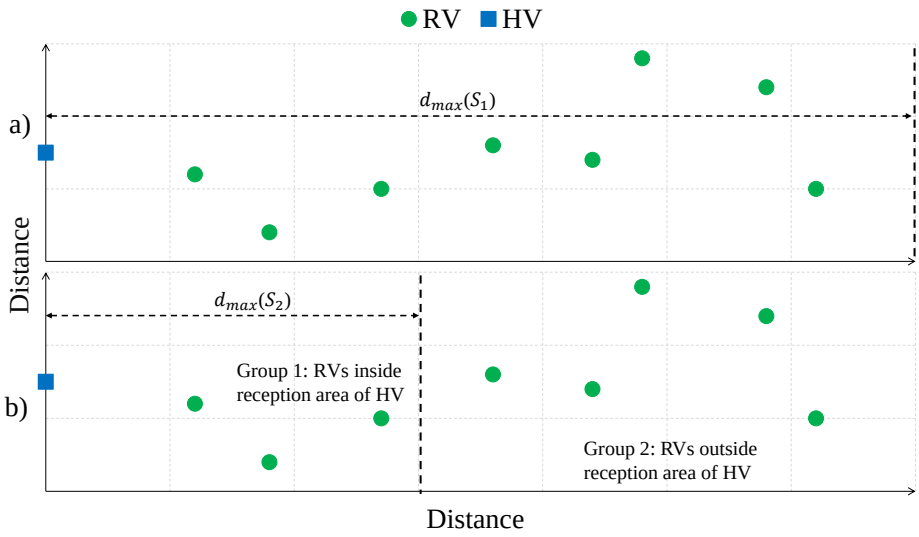


FIGURE 4.3: Change HV sensitivity S_i

After dividing the RVs into their respective groups, $N_{act}(S_i)$ and $N_{exp}(S_i)$ can be calculated. The value $m = |N_{act}(S_i) - N_{exp}(S_i)|$ represents the number of illegitimate vehicles. If $N_{act}(S_i) = N_{exp}(S_i)$, i.e., $m = 0$, all vehicles in $\mathbf{R}_i$ are marked as *honest*. In case of a discrepancy, i.e., $m \neq 0$, the HV investigates the message IDs of the BSMs received from these illegitimate vehicles and marks them as Sybil nodes.

If $N_{act}(S_i) < N_{exp}(S_i)$, the *malicious* vehicle is outside the reception area of the HV. Otherwise, it is inside. Based on this information, the algorithm changes the sensitivity to include or exclude the malicious vehicle in order to locate it. Only those BSMs originating from honest vehicles will be used.

4.3.3 *Simulation Setup*

We used the network simulator NS-3 to simulate the passive detection algorithm [60]. Of special interest are the two scenarios shown in Figure 4.4, in which the malicious nodes are at extreme positions, with respect to the Sybil nodes and the HV. Specifically, in the scenario of Figure 4.4 a) all Sybil nodes are positioned between the HV and the malicious node. The scenario shown in Figure 4.4 b) shows the other extreme, where the malicious node is closer to the HV than all Sybil nodes. In

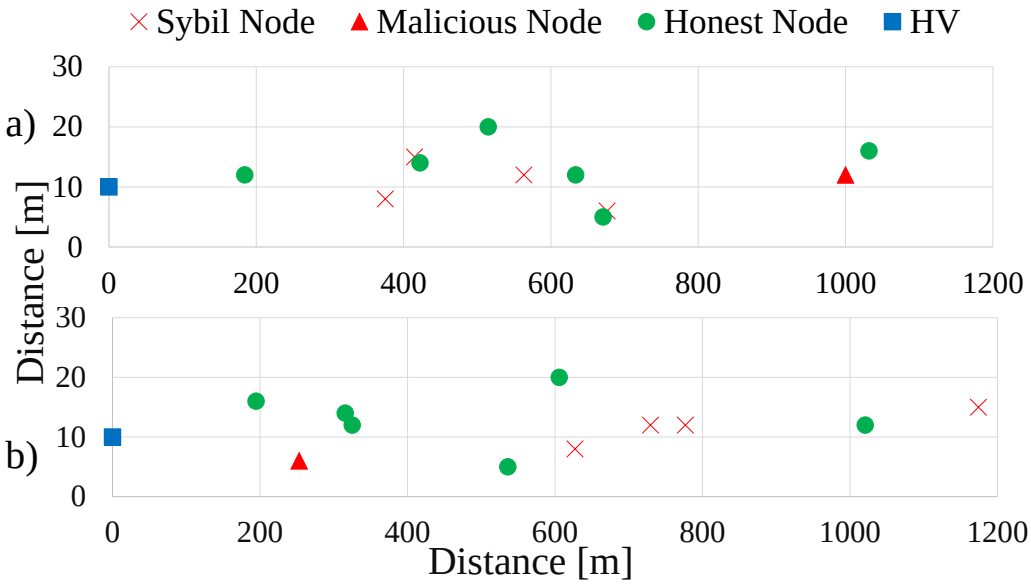


FIGURE 4.4: NS-3 simulation relative positions of malicious nodes with respect to HV and Sybil nodes

both scenarios the locations of honest and Sybil nodes were randomly selected. The reason for not including scenarios where the malicious node is in-between Sybil nodes is that such scenarios can be broken up into the previous two cases.

In the simulations, all nodes were initially configured to be within the reception area of the HV, moving with steady speed of 10 m/s on a straight multi-lane road.

The scenarios were set up to include one HV, six honest nodes and one malicious node, which acts as four Sybil nodes. The NS-3 simulation parameters are listed in Table 4.1.

TABLE 4.1: NS-3 Simulation Parameters

Propagation model	Friis Free Space
MAC protocol	802.11p
Antenna model	Omni-directional
Number of nodes	12 nodes (1 HV, 1 Malicious, 4 Sybil and 6 Honest)
BSM generation	10 BSM/s
Nodes speed	10 m/s
Transmitter power	16 dBm
Data rate	6 Mbps

Another experiment is conducted to calculate the HV's maximum reception distance for different sensitivity levels $d_{max}(S_i)$. The scenario demonstrated in Figure 4.5 consists of two vehicles representing a stationary HV and a moving RV. The

FIGURE 4.5: Determine HV's maximum reception distance

RV moves with constant speed of 10m/s, sends BSMs at a rate of 10 BSM/s and transmission power of 16 dbm, while the HV is configured to receive only. The PDR of the HV is calculated every second. For each Rx sensitivity level S_i, the maximum distance of reception $d_{max}(S_i)$ is determined when the PDR of the HV falls below 10%.

4.3.4 *Simulation Result Analysis*

Figures 4.6 and 4.7 show the relationship between the Rx sensitivity S_i of HV and the maximum distance of reception $d_{max}(S_i)$ for the scenarios depicted in Figure 4.4 a) and b) respectively. The colored symbols on left side of the figure represent different vehicles' positions. As shown in Figures 4.6 and 4.7 the HV can receive BSMs from

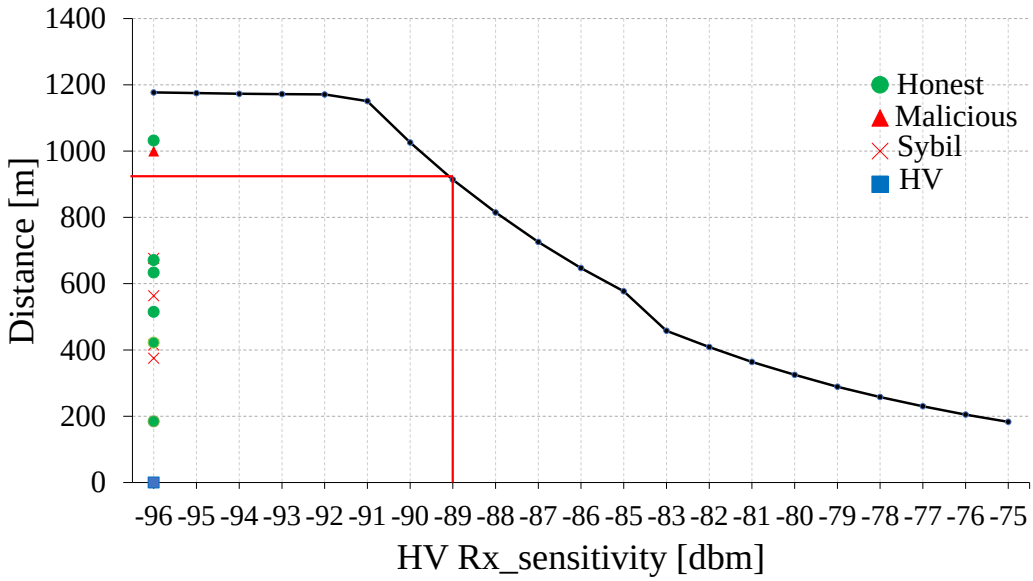


FIGURE 4.6: Relationship of Rx sensitivity and distance - Scenario of Figure 4.4 a)

all vehicles at default sensitivity $S_i = -96$ dbm since they were located within the HV reception area. However, when this sensitivity value decreases, the maximum distance of reception decreases as well, thus the number of vehicles from which BSMs were received is expected to decrease or at least stay the same. For example at $S_i = -89$ dbm, the HV should receive BSMs from 9 vehicles only, which are located below the red horizontal line in Figures 4.6 and 4.7, i.e., $N_{exp}(-89) = 9$. However the location of the malicious vehicle, whether located inside or outside the reception area of the HV, could affect the actual number of vehicles from which BSMs were received.

Figures 4.8 and 4.9 show the effectiveness of the passive detection algorithm for the scenarios described in Figures 4.4 a) and b) respectively. Graphs *Exp.Nodes* and *Act.Nodes* shown in Figures 4.8 and 4.9 represent the $N_{exp}(S_i)$ and $N_{act}(S_i)$ respectively, at different Rx sensitivity values. As mentioned earlier, at $S_i = -96$ dbm, set $\mathbf{R}_i$ includes all vehicles located in the reception area of the HV. To detect Sybil vehicles, the detection algorithm selects Rx sensitivity S_i to divide $\mathbf{R}_i$ into two groups. Then it calculates $N_{exp}(S_i)$ and $N_{act}(S_i)$. No Sybil detection occurs as long as there is no deviation between $N_{exp}(S_i)$ and $N_{act}(S_i)$, i.e., $m = 0$. Once a deviation is observed, this indicates that the HV is under attack. As can be seen in Figure

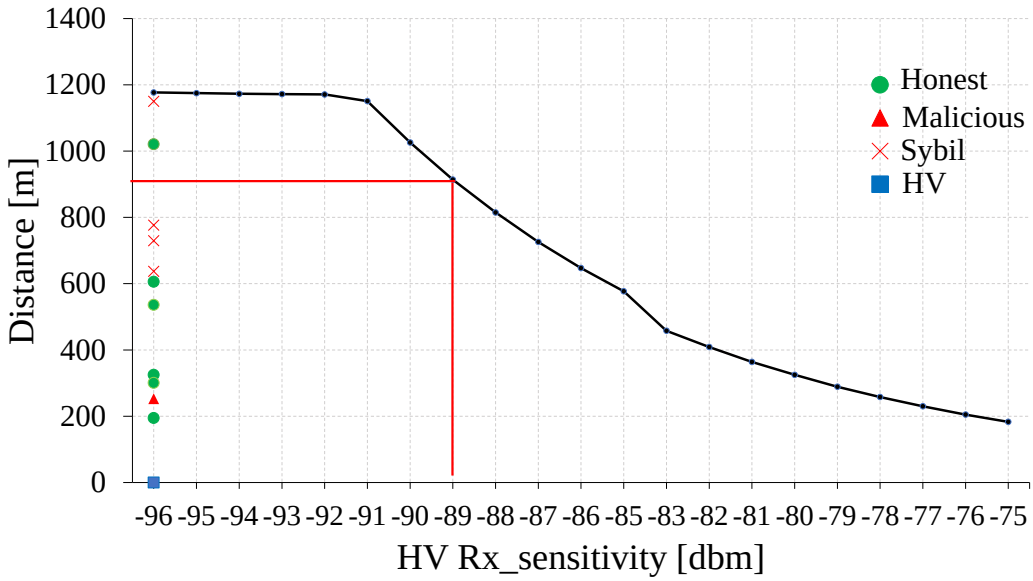
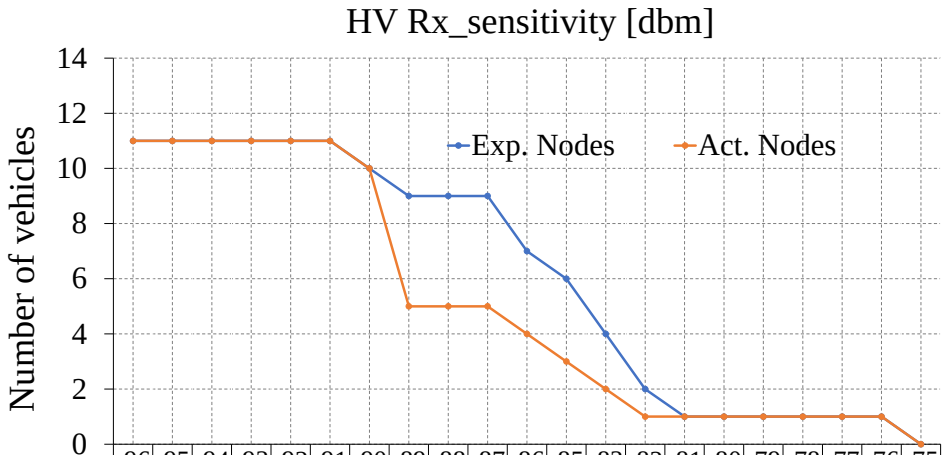


FIGURE 4.7: Relationship of Rx sensitivity and distance - Scenario of Figure 4.4 b)

4.8 there is no deviation between $N_{exp}(S_i)$ and $N_{act}(S_i)$ for all S_i values from -96 : -90 dbm. However, when S_i is tuned to -89 dbm using Equation 4.1, the $N_{exp}(-89)$ became greater than $N_{act}(-89)$ by 4, i.e., $m = 4$. It is shown in Figure 4.6 that at $S_i = -89$ dbm the corresponding $d_{max}(-89)$ = 900m. Accordingly, there were 2 vehicles outside of HV's reception area because their distances d_j from HV were greater than 900m. Thus, the calculated $N_{exp}(-89) = 9$. The malicious vehicle in this case was outside of the reception area of the HV as its distance from the HV is greater than 900m. Note that the malicious vehicle could send its BSMs using either the standard transmission power or manipulates it to be consistent with GPS coordinates of Sybil vehicles. In the latter case, it will be caught by the algorithm once a deviation from the standard power level is observed. While if the malicious vehicle uses the standard transmission power, the received power signals of the BSMs sent will be less than -89 dbm, which is too weak to be received by the HV. Thus $N_{act}(-89)$ will be equal to 5 only. In this case $N_{exp}(S_i) > N_{act}(S_i)$ means that the malicious node is outside of HV's reception area.

Now consider the scenario depicted in Figure 4.4 b), where the malicious vehicle is closer to the HV than all Sybil vehicles. As shown in Figure 4.7, when the sensitivity is tuned to achieve $d_{max} = 900$m at $S_i = -89$ dbm, the calculated



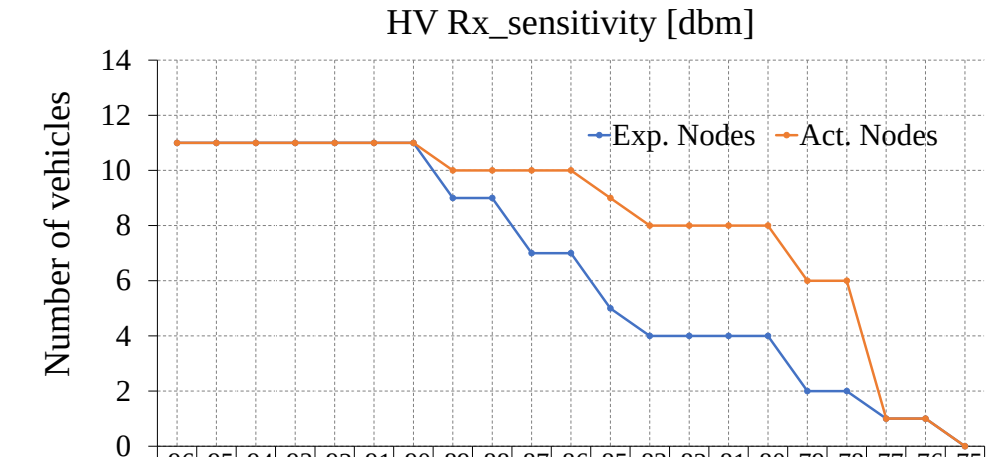
	-96	-95	-94	-93	-92	-91	-90	-89	-88	-87	-86	-85	-83	-82	-81	-80	-79	-78	-77	-76	-75
Exp. Nodes	11	11	11	11	11	11	10	9	9	9	7	6	4	2	1	1	1	1	1	1	0
Act. Nodes	11	11	11	11	11	11	10	5	5	5	4	3	2	1	1	1	1	1	1	1	0

FIGURE 4.8: Actual and expected number of vehicles - Scenario of Figure 4.4 a)

$N_{exp}(-89) = 9$ as there are 2 vehicles located at distances greater than 900m (outside the reception area). However, it results in actual reception from 10 vehicles, i.e., $N_{act}(-89) = 10$. This deviation observed at $S_i = -89$ dbm, $m = 1$, is due to one of these two vehicles that was now outside the HV's reception area. This implies the vehicle is a Sybil vehicle with fake position. The BSMs of the vehicle with fake position were actually sent from the malicious vehicle location that is still inside the HV's reception area. In this case $N_{exp}(S_i) < N_{act}(S_i)$, which indicates that the malicious node is inside the HV's reception area. Depending on whether $N_{exp}(S_i) > N_{act}(S_i)$ or $N_{exp}(S_i) < N_{act}(S_i)$, the algorithm calculates a new sensitivity to include or exclude the malicious vehicle in order to locate it.

4.4 ACTIVE SYBIL DETECTION

All passive Sybil detection algorithms, such as described in Section 4.1 and the above section, are of little use to detect Sybil attacks in dynamic power environments. The reason is that passive approaches, especially RSSI-based may not be precise enough, and do not work at all in dynamic power environments, since the sending power, which is needed for detection, is not known. In this section an active Sybil detection



	-96	-95	-94	-93	-92	-91	-90	-89	-88	-87	-86	-85	-83	-82	-81	-80	-79	-78	-77	-76	-75
Exp. Nodes	11	11	11	11	11	11	11	9	9	7	7	5	4	4	4	4	2	2	1	1	0
Act. Nodes	11	11	11	11	11	11	11	10	10	10	10	9	8	8	8	8	6	6	1	1	0

FIGURE 4.9: Actual and expected number of vehicles - Scenario of Figure 4.4 b)

algorithm is presented, which capable of detecting sophisticated Sybil attacks in a dynamic power environment.

4.4.1 *Impact of Forced BSM Queuing*

We introduce a Detection Packet (DP), which is a bogus packet of specified duration, to check if selected nodes queue their BSMs. Queuing is the result of the node having no medium access at the MAC layer. Figure 4.10 shows the impact of forcing BSMs to be queued for 500ms by means of a DP. The bars in Figure 4.10 represent the number of BSMs received by the HV from four nodes, each emitting BSMs every 100 ms. Specifically, these four nodes were stationary, one node was positioned 80m from the HV, while the other three nodes were 150m from the HV. The Figure can be divided into three intervals, the closed time interval $[0, 1.1]$, the open time interval $(1.1, 1.7)$, and interval $[1.7, 2.6]$. During the first interval the HV received 4 BSMs every 100 ms, whereas no BSMs were received during the second interval, due to the DP. Note that this DP was sent with a power of 1 dbm to affect only the nearest node, i.e., RV1. As a result, RV1 queued 5 BSMs as it could not access the media, and obviously no BSM was received by the HV. In the third interval, the 5 queued BSMs of RV1 were sent in a burst. The investigation of the message IDs of the BSMs

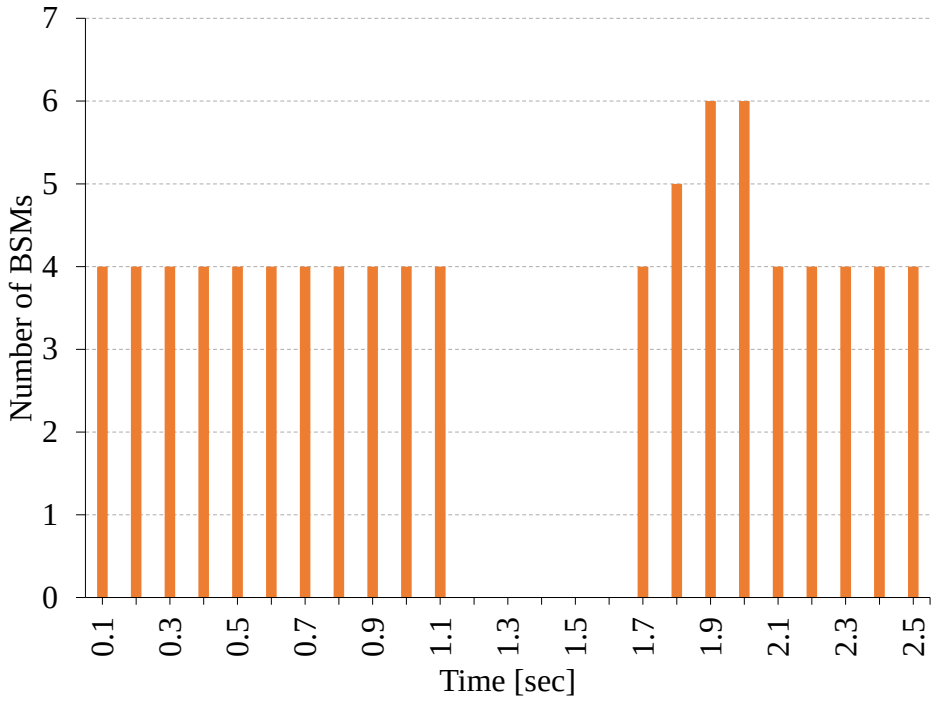


FIGURE 4.10: Field experiment with a DP duration of 500ms

received by the HV confirmed that the bust of RV1 was among them. The other three nodes outside of the range of the DP sent their BSMs normally. However, these BSMs collided with the DP, and were thus not received.

The above experiment shows that the HV could in fact tune the power level of the DP to affect and observe a specific node, in this case only RV1. However it should be pointed out that long detection packets are like an HV-induced DoS attack [61], which motivated the investigation of shorter DP durations. Furthermore, it should be noted that due to the lower power of the DP, the area affected is reduced.

Figure 4.11 shows the impact of forced queuing of BSMs with a 50ms DP, displaying the cumulative inter-arrival times of 10 BSMs. In this particular experiment the DP caused the 5^{th} BSM to be delayed by approximately 30ms.

4.4.2 *Active Sybil Detection Algorithm*

The proposed algorithm shown in Figure 4.12 is capable of detecting sophisticated Sybil attacks by considering two cases. In the first case the Sybil nodes are assumed to be positioned between the HV and malicious node, whereas in the second case

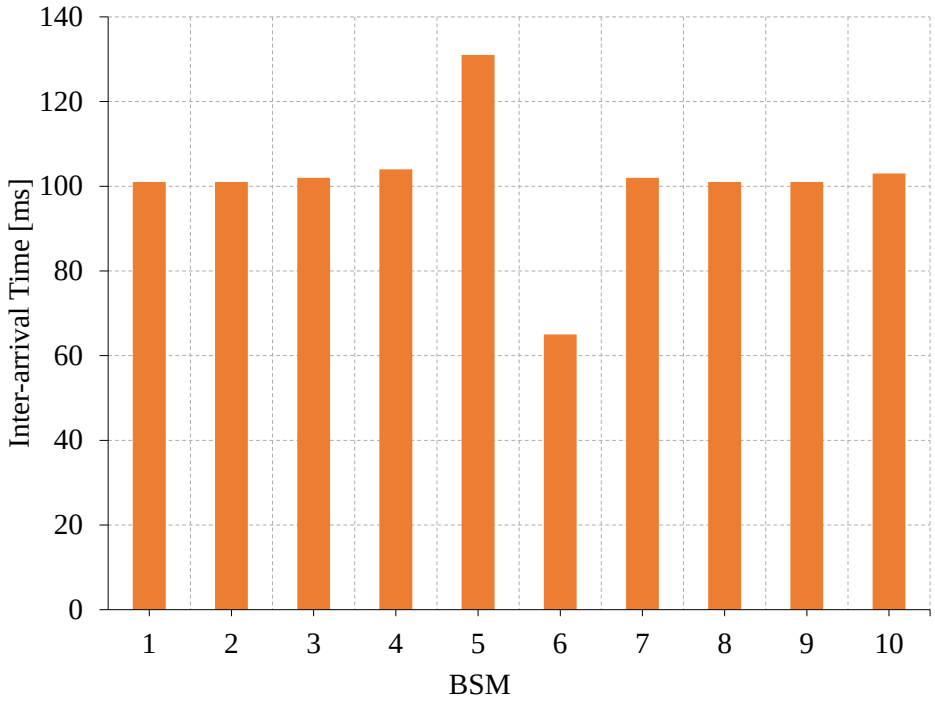


FIGURE 4.11: Cumulative BSM inter-arrival times for a DP of 50ms

the malicious node is between the HV and any Sybil nodes. As mentioned before, all other cases can be derived from these two cases. Thus, if the malicious node has Sybil nodes on either side, one can partition the Sybil nodes into two groups, i.e., those between the HV and malicious node, and those after the malicious node.

The detection algorithm works as follows. Upon receiving a message from a suspecting vehicle, the algorithm calculates the distance between the HV and the suspected vehicle, using the GPS coordinates, which may be correct, or spoofed, as in the case of a Sybil node. Next, the transmission power P_{snd} to be used for the DP is determined using

$$P_{snd} = S \times d^2 / G \tag{4.6}$$

where S is the receiver sensitivity of the suspected vehicle, d is the distance between the HV and the suspected vehicle, and G is the gain, as computed by Equation 5.2. Now the HV sends the DP of duration τ. The exact value to be used for τ will be discussed later. The DP forces nodes that expect to send a BSMs during the time that overlaps with the DP to be delayed, as their MAC layer access is blocked. If the

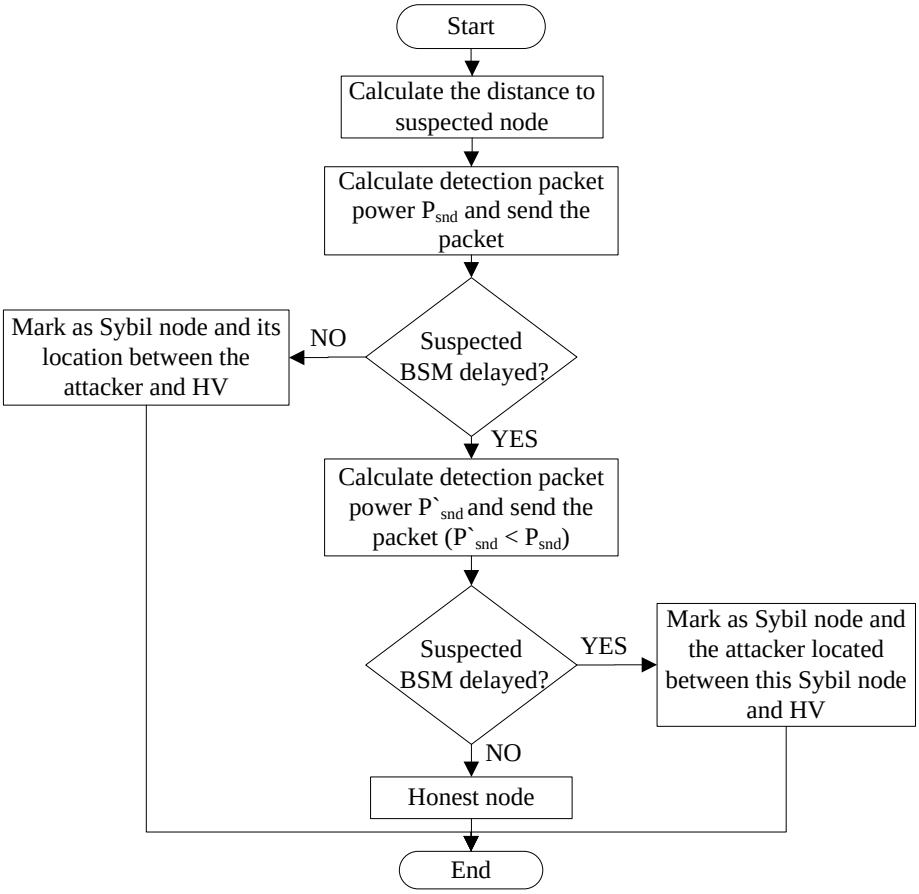


FIGURE 4.12: Active Sybil detection algorithm

DP does not cause a delay, then this node is marked *Sybil* and it is located between the attacker and the HV. Otherwise, a new P'_{snd} is computed that will exclude the suspected node, i.e., $P'_{snd} < P_{snd}$. Thus, the DP sent with that power will not interfere with the suspected node. If the DP does not cause a delay, then this node is marked *honest*. Otherwise it is a Sybil, and the attacker is located between the HV and the Sybil. Based on the status of a node, *honest* or *Sybil*, the HV can consider or reject BSMs from this node respectively.

4.4.3 *Field Experiments*

The feasibility of the Sybil detection algorithm was tested using field experiments. One vehicle representing the HV and one vehicle representing a malicious node

acting as four Sybil nodes were equipped with LocoMate Classic OBUs from Arada Systems [20]. The malicious OBU was configured to act as 4 different OBUs (4 Sybils) with different message IDs and GPS locations. All OBUs transmitted BSMs every 100ms on safety channel CS172, using a transmission power of 23 dBm. The OBU in the HV executed the active detection algorithm capable of sending DPs with different transmission powers. The experiments were conducted in a controlled configuration, where the vehicles were stationary. The HV sent detection packets using a power of $P_{snd} = 1$ dBm for different DP durations $\tau = 25$, 50, 75 and 100ms. The HV was placed 80 meters from the malicious node. The rational for conducting the experiments with stationary, rather than moving vehicles, was to eliminate the impact of external influences, such as changing road geometry (e.g., curves), elevation changes, and unrelated road traffic, as no dedicated test site was available. Table 5.1 summaries the parameters used in the experiment.

TABLE 4.2: Sybil Detection Field Experiment Parameters

OBU Model	Arada Systems LocoMate Classic
Number of OBUs	2 (1 HV and 1 malicious)
Test range	Straight two-lane road
Distance: HV to malicious	80 m
Vehicles speed	0 m/s (Fixed)
Tx power & Data rate	23 dBm, 3 Mbps
BSM generation	10 BSM/s
Channel	Safety Channel 172
DP power & data rate	1 dBm, 3Mbps
Delay sensitivity threshold δ_t	25ms

As indicated above, of special interest is the position of the malicious node with respect to its Sybil nodes. Two scenarios are shown in Figure 5.3, in which the malicious nodes are at the extreme positions with respect to the Sybil nodes and the HV. Specifically, in the scenario of Figure 5.3a) all Sybil nodes are positioned between the HV and the malicious node. The scenario in Figure 5.3b) shows the other extreme, where the malicious node is closer to the HV than all Sybil nodes. Recall that scenarios where the malicious node is in-between Sybil nodes can be broken up into the previous two cases. In both cases the first Sybil node is at distance 80m and the DP transmission spans to distance 100m.

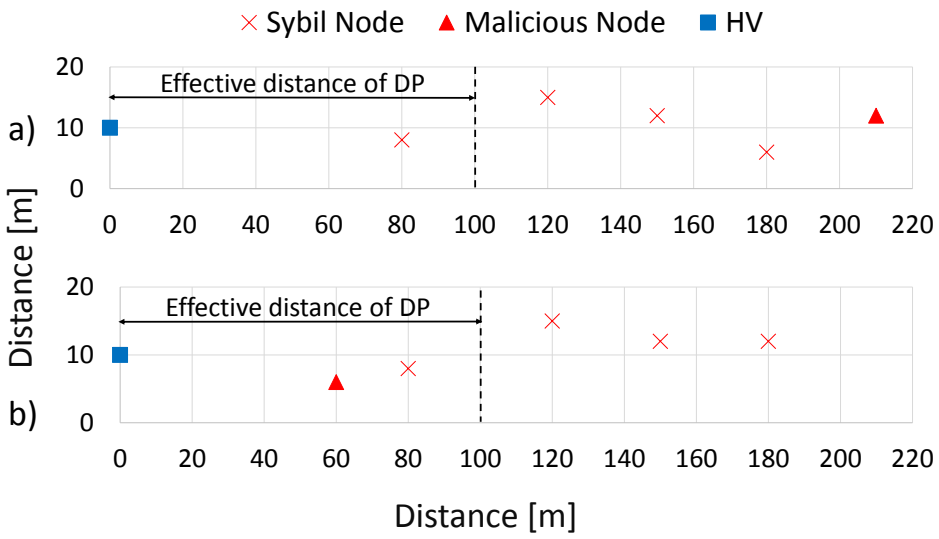


FIGURE 4.13: Extreme position of malicious nodes

4.4.4 *Experimental Result Analysis*

The Sybil detection algorithm is based on the capability of detecting a delayed BSM transmission of a suspected node as the result of a DP. Several scenarios are possible. Either the DP will overlap with the time interval of the intended BSM transmission or not. If it does not, the transmission time of a BSM is unaffected by the DP, and thus there is no delay. If however it does overlap, then it will result in a BSM delay or a collision. Collisions occur if the BSM was sent from a node outside of the range of the DP. Let δ_t be the minimum delay time that will constitute a recognized delay. δ_t is thus a parameter that allows tuning the sensitivity of delay detection, e.g., to account for MAC access delays due to other network activity. Any BSM delay shorter than δ_t is ignored. Let q be the probability that DP is sent in such fashion that it overlaps and results in a delay of at least δ_t. If the DP has a duration of $\tau = 100\text{ms} + \delta_t$, then a delay is recognized with high probability, approaching 1. However, such long DP may be too invasive. Thus, lower DP durations are desirable, but they may lead to unrecognized delays.

Figure 4.14 shows the probability q of delay detection for DPs with different τ for two DP transmission precisions F1 and F2. Precision relates to how precise one can time the transmission of the DP to delay a BSM. F1 was the precision of our DP generation and transmission mechanism, a jamming application developed

by Arada for use in our jamming related research such as [61]. Imprecision was due to variable startup times of the jammer, as the result of process setup and switching times of the Arada LocoMate Classic's operating system. The precision of F2 was higher as the result of implicit timing manipulation. In either case shorter DP durations resulted in lower delay recognition probabilities, but delay recognition was significantly higher for F2.

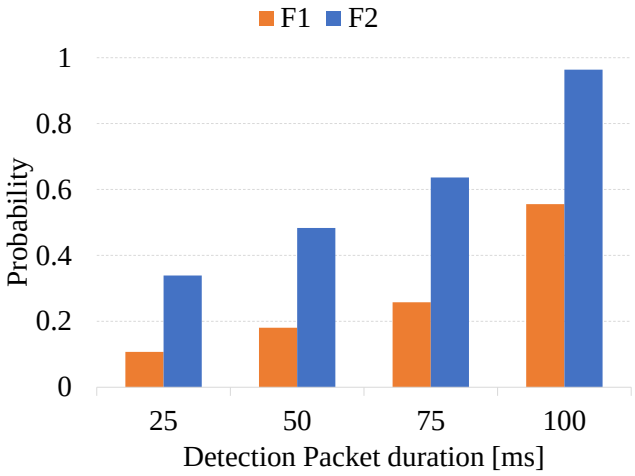


FIGURE 4.14: Probability of delay detection

Effective delay detection in the context of DSRC Safety Applications implies that at least one BSM delay has to be detected. For each BSM there is a probability of $1 - q$ that detection will fail. Every 100ms, the BSM transmission period, there is another chance to detect delay. Assume that N BSMs are considered for delay detection. Then the probability ϵ of not observing a delay of any of the N BSMs is

$$\epsilon = (1 - q)^N \tag{4.7}$$

As N grows larger, i.e., as more BSMs are considered, the probability of missing all delays decreases exponential.

Field experiments were conducted, consisting of 10 repetitions with 20 DP for different τ durations. The results for precisions F1 and F2 are shown in Figures 4.15 and 4.16 respectively, which show the trade-off space between N, ϵ and τ. The observations were over 20 BSMs, implying time intervals of 2 seconds at the 10

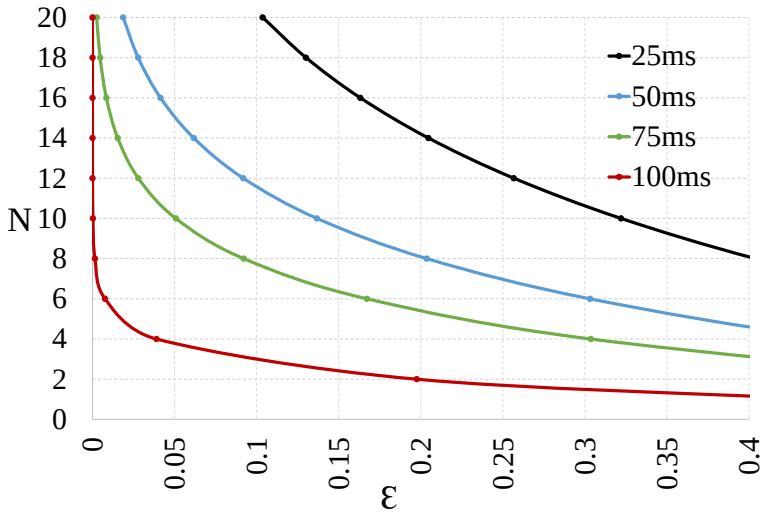


FIGURE 4.15: Low-precision scenario F1

BSM/s rate. This represents a scenario in which cars drive with 3 seconds separation, and assuming a reaction time of 1 second, thus leaving 2 seconds, or 20 BSMs, to detect at least one delay. Probabilities ϵ (of not observing a delay) were considered for ϵ in $[0, 0.4]$. The figures show that shorter DPs result in larger probabilities of not observing any delay. More importantly, the figures show the impact of the DPs placement precision. Specifically, in Figure 4.16, where each DP was placed more precisely to delay a BSM, significantly fewer BSMs were required to detect delays with smaller error probability ϵ. Especially for F2 very high delay detection probabilities could be archieved with few BSMs, e.g., even in the case of $\tau = 25$ms, it took only 6 BSMs to achieve an ϵ of 0.1, or alternatively a delay recognition probability of 90%. Very high recognition probabilities could be achieved for all τ when N was larger. Specifically, close to 100% detection was achieved for $\tau = 25, 50, 75$ and 100ms for $N = 16, 10, 6$ and 2, respectively. This shows that even with shorter DP Sybil detection is highly effective if one considers multiple BSMs in the detection algorithm.

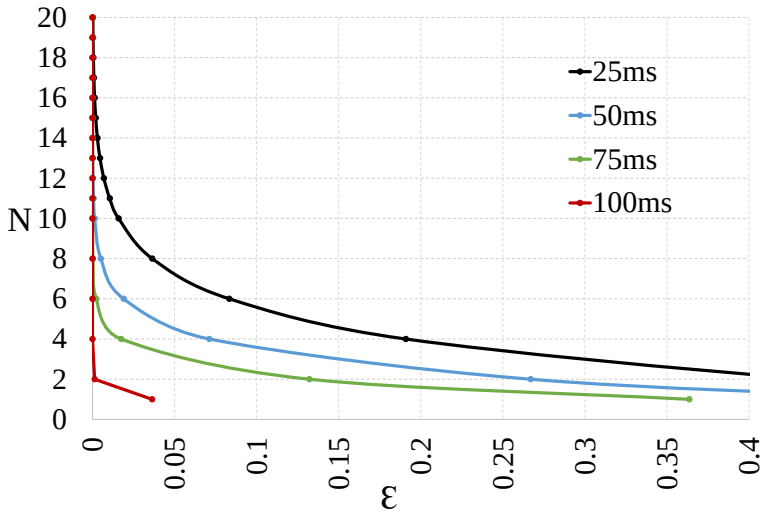


FIGURE 4.16: High-precision scenario F2

4.5 CONCLUSIONS

This chapter first presented a passive algorithm to detect Sybil attacks in a static power environment, which can cause serious problems for DSRC safety applications using voting schemes. The passive algorithm has two phases. First, it checks the power field in the BSM header to catch Sybil vehicles that might manipulate the transmission power. Second, it applies an algorithm that varies the Rx sensitivity, to calculate the number of vehicles observed. Any deviation from the expected to the observed number of vehicles may indicate a Sybil attack. Result shows that the passive algorithm is only suitable for static power environment. However, it is ineffective in dynamic power environment. Therefore, an active algorithm to detect Sybil attacks is proposed, which is suitable for a dynamic power environment. This algorithm can detect Sybil nodes by using a Detection Packet to investigate queuing delays of nodes. In order to minimize the intrusiveness of these DPs, delay detection was spread over multiple BSMs. Analysis and field experiments on the effects of the number of BSMs, DP duration, and time placement accuracy revealed that DP placement and duration have the largest impact on Sybil detection effectiveness.

CHAPTER 5

An Enhanced Active Sybil Detection Algorithm and its Impact on Reliability of Safety Applications in VANET

This chapter focuses on developing solutions to increase reliability of safety applications in the presence of faults and attacks. It is based on the work in [24], where an active detection algorithm was presented that is capable of detecting different attack scenarios using detection packets, which in turn helped improve the resilience of safety applications to Sybil attacks launched by rouge nodes. Extending the approach in [24], the main contributions of this chapter are an 1) analysis of the impact of the frequency and duration of detection packets on Sybil node detection, and 2) the presentation of an enhanced algorithm using these two metrics. The impact of the algorithm on DSRC safety application reliability is shown in the context of the fault models in [71] and [72]. To put these fault models into perspective, a description of fault models is presented next.

5.1 FAULT MODEL

In the context of fault-tolerance, a *fault* is a physical defect or flaw that occurs in a hardware or software component [42]. This flaw can produce an *error*, which is the manifestation of the fault. An error in turn can lead to a *failure* of a component or system. a more elaborate description of the relationship between fault, error, and failure can be found in [42] and [62].

In research such as presented in [63, 64, 67], all faults were assumed to be worst case. However, faults may show different behavior, which in turn may have different consequences for a system. The result of such worst case assumptions is that systems might be over-designed. The term *Fault Model* was introduced to describe taxonomies of faults. It shows the distinction between faults based on their behavior, and helps to identify the potential impact on the system and to find

appropriate strategies against these faults. Figure 5.1 shows a hierarchy of fault models.

Generally speaking, faults can be partitioned into either benign or malicious faults [65], as shown in the second level of Figure 5.1. Benign faults are self-evident and can be diagnosed globally by all non-faulty nodes. A crash fault is an example of this fault type. On the other hand *Malicious* faults are not self-evident. They may behave in different ways by some nodes in a redundant system. These faults are also known as Byzantine faults after the Byzantine General's Problem [66, 67]. Such faults are hard to deal with as they behave arbitrarily. In [67] they stated that $N \geq 3m + 1$ nodes are needed to deal with m malicious faults.

The third level of the figure shows the model proposed in [68]. This model categorized malicious faults into symmetric or asymmetric faults. A *symmetric* fault assumes that all nodes received the same faulty value or all receive no value, whereas an *asymmetric* includes all other behavior. Asymmetric faults are Byzantine faults as described in [67].

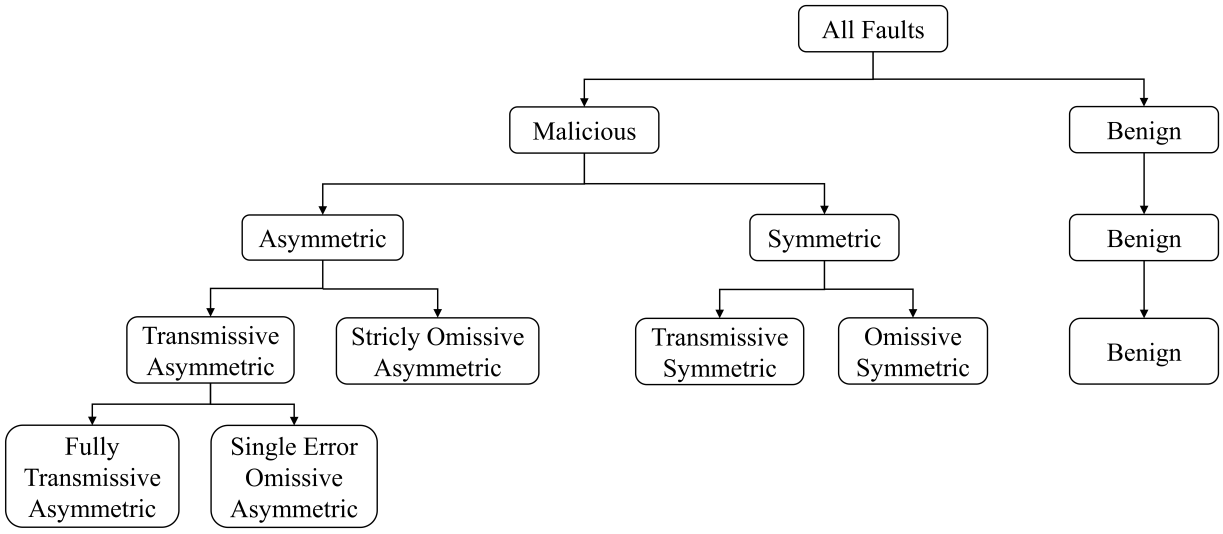


FIGURE 5.1: Fault taxonomy

The faults model of [71], shown in the fourth level of Figure 5.1, is known as Omissive/Transmissive Hybrid Five-Mode (OTH-5) fault model. This model broke down the symmetric faults into two groups. The first group, *Omissive Symmetric* faults are caused when all receiving nodes received no value from the sender. How-

ever, this type of faults are not globally diagnosed, as the receiving node is unable to confirm if the omission was detected by all other receiving nodes. Thus it can not be considered a Benign fault. The second group, *Transmissive Symmetric* faults occur when the same erroneous value is delivered to all nodes. It should be noted that this fault is considered as symmetric fault in [68].

Asymmetric faults in this model are divided into two categories as well. *Strictly Omissive Asymmetric* fault is the first category, in which all non-faulty nodes either receive a correct value or no value. Thus no wrong value is sent. The other category is a *Transmissive Asymmetric* fault, which is the classic Byzantine fault.

The final level of Figure 5.1 depicts another refining of transmissive asymmetric fault presented in [72]. This model is also known as Omissive/Transmissive Hybrid model with six fault modes (OTH-6). The transmissive asymmetric faults are partitioned into two new groups, *Fully Transmissive Asymmetric* and *Single Error Omissive Asymmetric* faults. The latter means that some nodes do not receive a value, while other nodes receive a single erroneous value. The erroneous value is the same for all non-faulty nodes that received a value form a particular faulty node.

5.2 Enhanced Active Sybil Detection Algorithm (EASDA)

The proposed detection algorithm is based on the ability to detect the delay of a BSM caused by the DP [24]. This DP has a aforementioned predefined duration τ. The time between two consecutive DPs is specified as period T. Detection is based on the observation in the targeted vehicle's queue, i.e., if it queues its BSMs or not. In this algorithm Sybil attacks can be detected by taking into account two cases. The assumption in the first case is that the Sybil vehicles are placed between the HV and the malicious vehicle, while in the second case the malicious vehicle is placed between the HV and any Sybil vehicles. From these two cases we can deduce all other cases. Thus, if the malicious vehicle is in-between Sybil vehicles, we can divide the Sybil vehicles into two groups, those between the HV and malicious vehicle, as

in the first case, and the rest form the scenario of the second case. These scenarios will be further discussed below in the description associated with Figure 5.3.

The detection algorithm is shown in Algorithm 1. In order to keep track of illegitimate (Sybil) vehicles, a Blacklist (BL) is defined, which is initially empty. The BL stores the vehicle IDs of detected Sybil nodes. Once a BSM from a suspecting vehicle is received, the algorithm calculates the distance d, which is the distance between the HV and the suspected vehicle. The GPS coordinates included in the received BSM, which are probably spoofed, are used to calculate d. The variables τ and T need to be determined (at line 4 of the algorithm). The values of the variables are affected by the number of vehicles in the neighborhood and distance d. Furthermore, recall that large values of τ are highly disruptive and can be seen as a self-induced DoS. As a result, it is desirable to have shorter durations, which however reduces the probability of detection. This can be compensated by increasing the frequency of sending a DP, i.e., a reduction in T. How τ and T are determined and the trade-off associated with their values will be discussed later in Section 5.4. The time available for the algorithm to detection a Sybil node will be the time it takes the HV to reach the suspected node, minus the reaction time. This time is $T_{detect} = d/V_{HV} - T_{react}$. Any detection after that such duration will be too late. The transmission power P_{snd} of the DP is computed using

$$P_{snd} = S \times d^2/G \tag{5.1}$$

where S is the receiver sensitivity of the suspected vehicle, which is assumed to be known (Subsection 4.2), and G denotes the gain. This gain is calculated as in [59] to be

$$G = G_{snd} \times G_{rcv} \times \lambda^2/(16\pi^2) \tag{5.2}$$

where λ is the wavelength of the radiation, G_{snd} and G_{rcv} are the send and receive gains, which are assumed to be known. The DP is sent with transmission power P_{snd} and for duration τ (in line 5 of the algorithm). The two scenarios shown in Figure 5.2 can occur. Either DP does not interfere with the timing of BSM transmission, shown in Figure 5.2a), or the DP causes BSMs from vehicles that intend to send during

Algorithm 1 Enhanced Active Sybil Detection Algorithm

```
Initialize BL= ∅;
START: Receive BSM from a suspecting vehicle
Calculate distance d;
Determine τ and T;
Calculate P_snd and send DP(P_snd);
if (BSM delay <δ_t) then
   Mark vehicle as Sybil;
   Add vehicle's ID to BL;
else
   Calculate P'_snd and send DP(P'_snd);
   if (BSM delay <δ_t) then
      Mark vehicle as Honest;
   else
      Mark vehicle as Sybil;
      Add vehicle's ID to BL;
   end if
end if
if  Detection uncertainty ϵ > specified value then
   Goto: START;
else
   Stop Algorithm;
end if
```

the time that overlaps with the duration of DP to be queued, as the medium is blocked. This scenario can be seen in Figure 5.2b), where BSM(i+1) is delayed. The queuing delay is the sum of the BSM's delayed medium access due to the DP and possible delay due to contention of other BSMs. Contention delay cannot be estimated precisely as it is nondeterministic and may increase with traffic density. We therefore declare a *Delay Sensitivity threshold* δ_t to be a tunable minimum time threshold for a considered delay, i.e., a delay less that δ_t is interpreted as no delay. Thus, the scenario in Figure 5.2b) is the boarder case for Sybil detection. Parameter δ_t is used to adjust the sensitivity of the detection algorithm. For the sake of simplicity we will consider it constant in the discussion to follow.

If the DP delays less than δ_t, or if no delay occurred, then this vehicle is marked as *Sybil* and its position must be between the malicious node and the HV. Otherwise, a new P'_{snd} is computed that will cause the suspected vehicle to be out of range. Thus, the DP sent with that power would not be received by the suspected vehicle.

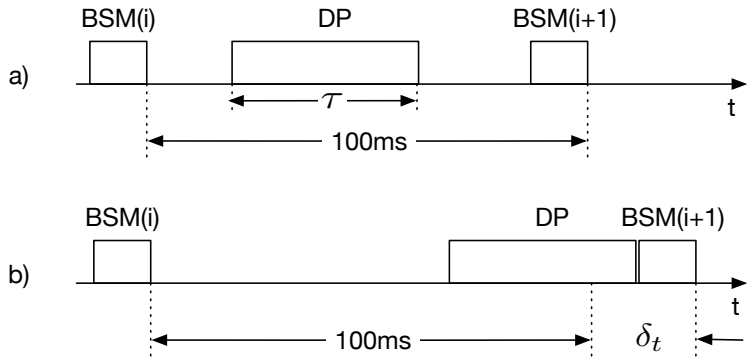


FIGURE 5.2: BSM delay due to DP

If the delay caused by the DP is less than δ_t or no delay occurred, then this vehicle is marked *Honest*. Otherwise it is a Sybil node, and the malicious node is positioned between this Sybil and the HV. Based on the status of a vehicle, i.e., Honest or Sybil, the HV can accept or reject BSMs from this vehicle respectively. In the latter case the Sybil is added to the blacklist. The BL can be shared with other participants, e.g., by sending it to the RSU, which in turn can relay this information with other RSUs [70]. This could be used by an RSU to inform upcoming traffic about Sybil vehicles. Such process implies that a reporting HV is correctly authenticated by the RSU to avoid malicious nodes framing honest nodes as Sybil nodes. Algorithm 1 terminates when a specified detection probability is achieved, i.e., if detection uncertainty ϵ is below a certain threshold, as will be describe in Subsection 5.4.2.

5.3 SAFETY APPLICATION RELIABILITY IN PRESENCE OF EASDA

The impact of τ and T on safety application reliability will now be discussed in the context of the fault model in [71]. Two fault scenarios are considered.

The first assumes undetected Sybil attacks. As a result of non-detection falsified BSMs are used, which constitute value faults for the safety application. In [71] such fault type is called *transmissive symmetric*, which is a value fault in which a false BSM is received by all vehicles. If however the false BSM was not received by some

vehicles, then this constitutes a *Single Error Omissive Asymmetric* fault, described in [72].

The second scenario addresses the case where BSMs are delayed by DPs to the point where the information is deemed outdated. According to [45] this time-to-live of a BSM should be no more than 500ms. BSMs that exceed the time-to-live should be discarded. In the context of the fault model in [71], this timing fault effectively causes an *omissive symmetric* fault. Specifically, in [71] an omissive symmetric fault implies that no value was received by any node. In our case the result is that no value is used by any node, as it is discarded due to being outdated.

The safety application will fail in either of the following two scenarios: 1) the Sybil detection algorithm fails to detect the attack or 2) BSMs containing crucial information, such as an event, are discarded due to the second scenario. In reliability analysis this can be represented by a series reliability block diagram [42], and thus the safety application reliability, $R_{app}(t)$, can be expressed as $R_{app}(t) = R_1(t)R_2(t)$, where $R_1(t)$ is the detection probability, and $R_2(t)$ is the probability of receiving at least one BSM containing an event before it is too late to react.

5.4 EXPERIMENTS AND RESULTS

5.4.1 *Field Experiments*

The feasibility of the Enhanced Active Sybil Detection Algorithm (EASDA) and its impact on the reliability of DSRC safety applications was examined using field experiments. Four vehicles were equipped with LocoMate Classic OBUs from Arada Systems [20] representing one HV, two RVs and a malicious node acting as four Sybil nodes with different GPS locations and message IDs. The EASDA was installed on the OBU in the HV, which thus allowed it to send DPs with different transmission powers, durations, and periods. All OBUs were sending BSMs using a transmission power of 23 dBm, a data rate of 3 Mbps, and the standard BSMs transmission rate of 10 BSMs/s on safety channel SCH 172. After extensive experimentation with the setup above, a sensible value for the delay sensitivity threshold δ_t was determined as 25ms. The position of vehicles can be seen in Figure 5.3 for the two

aforementioned extreme locations of the malicious node. It should be noted that all nodes were stationary in a controlled configuration. The reason behind conducting the experiments with stationary nodes instead of moving nodes was to isolate the experiment from any external influences, including changes in elevation, unrelated traffic or road layout, as no dedicated filed site was available. The experiment parameters are summarized in Table 5.1.

TABLE 5.1: Field Test Parameters

OBU Model	Arada LocoMate Classic
Number of OBUs	4 (1 HV, 2 RV, 1 Malicious)
Test road range	Straight two-lane road
Distance: HV to RV	80 m
Vehicles speed	0 m/s (Fixed)
Tx power & Data rate	23 dBm, 3 Mbps
BSM generation rate	10 BSM/s
Channel	SCH 172
DP power & Data rate	1 dBm, 3Mbps
DP durations τ	25, 50, 75 and 100ms
Delay sensitivity δ_t	25ms

In the scenario in Figure 5.3a) all Sybil nodes are positioned between the malicious vehicle and the HV. Figure 5.3b) shows the other scenario, where the malicious vehicle is the closest to the HV. The closest suspected Sybil vehicle is located 80m from the HV in both scenarios. The DP transmission power was computed using Equation 5.1, resulting in a coverage distance of 100m.

Recall that the EASDA relies on the ability to cause the suspected vehicle to delay its BSMs as a result of a DP. Any BSM delay time shorter than δ_t, e.g., less than 25ms, will not be considered to be delayed. The DP can overlap with the time interval of an intended BSM transmission, which in turn will cause a delay of that BSM. If it does not overlap, then no delay will occur. However, collisions may occur if a node located outside the range of the DP broadcasts a BSM. Such scenario is called a *hidden terminal* or *hidden node* problem [69]. Field tests were conducted consisting of 10 experiments, each consisting of 20 DPs with different durations τ and periods T. Specifically, $\tau = 25, 50, 75$ and 100ms were chosen to study the impact of the durations on delay detection. The reasons for the range of values

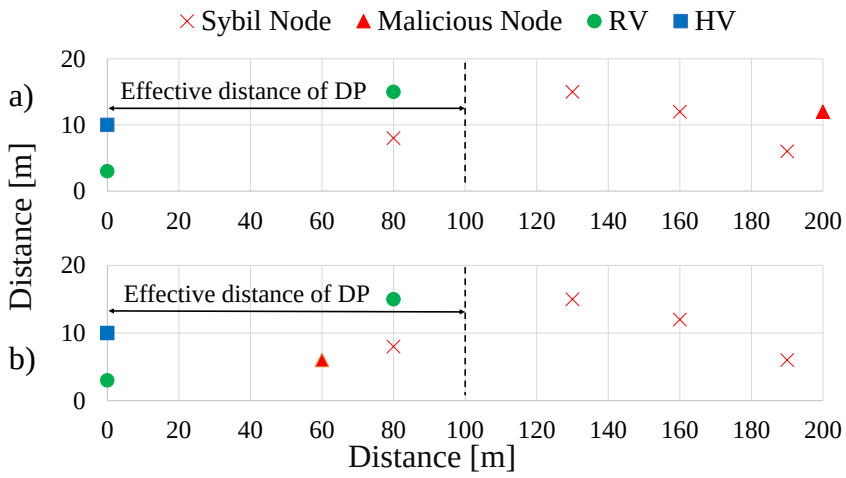


FIGURE 5.3: Relative position of malicious node to Sybil nodes

for τ will be discussed below. Obviously, long DP durations have high probability of delay detection, especially if τ exceeds the BSM spacing, e.g., by more than δ_t. However, such long DP may be too invasive, suggesting the investigation of shorter DP durations, although those may not lead to recognized delays, and thus fail to detect a Sybil node for that specific DP. Thus, one should look at detection using shorter τ over multiple BSMs with DP rate T.

The actual values of T will have an impact of the delay detection probability. For example, increasing the frequency of sending short DPs will also increase the probability of detection. As indicated before, T_{detect} is the maximum time available to the EASDA for Sybil detection. Thus the values for τ and T should be analyzed in the context of T_{detect}, the duration of time left before t_{react}. The safety application reliability is therefore directly linked to this detection probability.

5.4.2 *Detection Probability*

To determine the detection probabilities for different parameters the algorithm was tested under conditions similar to that described in Subsection 4.4.3. Recall that q is the probability that a DP overlaps with the intended transmission time of a suspected BSM in such a way as to cause a delay greater than or equal to the minimum recognition delay δ_t. Figure 5.2b) depicted such case. Assume δ_t =25ms, as indicated in the table. Furthermore, assume durations were $\tau = 25, 50, 75$, and

100ms, denoted by $\tau_{25}, \tau_{50}, \tau_{75}$ and τ_{100} respectively. If DPs are sent without precise time placement, i.e., their transmission times are random, then detection probability q for a single DP sent out to target a BSM from a specific vehicle can be calculated as $q = 0.2, 0.4, 0.6$, and 0.8 for $\tau_{25}, \tau_{50}, \tau_{75}$ and τ_{100} respectively. This is calculated using the parameters given and relating them to Figure 5.2b), i.e., for a given τ_x the detection probability $q(\tau_x) = \tau_x/(100ms + \delta_t)$.

Assume that the separation time between vehicles is 3s. With an assumed reaction time of 1s a driver has 2s left to react to a safety application alert. Given the standard transmission rate of 10 BSM/s a total of 20 BSMs can be considered for potential BSM delays during that time. To experimentally measure q for a single DP, field tests were conducted for different τ. As in Subsection 4.4.3, the field tests consisted of a series of experiments with 20 DPs each, to mimic BSMs over 2s. The results are shown in Figure 5.4, where the calculated q can be compared with the minimum, maximum and average values as derived from 7 experiments of 20 DPs for each τ. Whereas we conducted a total of 120 experiments, we intentionally did not average over a large number of experiments, as we were interested in seeing the impact of small samples when calculating q. Inspecting individual experiments we could not observe any obvious patterns for delay detection as the result of DPs. This was even the case when sending DPs periodically for specific T. As expected, shorter durations τ result in lower delay detection probabilities q than longer durations. For example, the average q for $\tau = 25ms$ and $100ms$ increased from 0.17 to 0.83 respectively.

Sybil detection of EASDA is based on detecting at least one BSM delayed by δ_t or more. Since q is the probability of detecting a delay as the result of one DP, $1 - q$ is the probability that either the DP does not overlap with the BSM, as shown in Figure 5.2a), or it overlaps, but not enough to cause a delay of δ_t. Every time a DP is sent, there is a chance for delay detection. If the number of BSMs considered for delay detection is N, then the probability of not detecting a BSM delay with any of those N DPs is Equation 4.7, restated as

$$\epsilon = (1 - q)^N \tag{5.3}$$

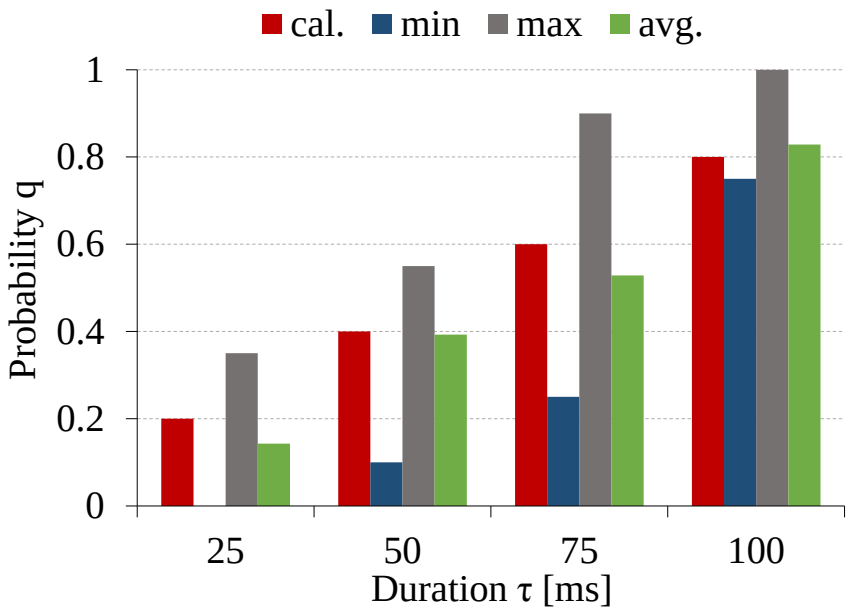


FIGURE 5.4: Probability of delay detection for different τ

Probability ϵ is the unreliability of the Sybil detection algorithm. Thus detection probability $R_1(t)$, defined in Section 5.3, is equal to $1 - \epsilon$. As N increases there are more chances to cause a BSM delay, and therefore the probability of not observing at least one delay decreases exponentially. The following questions arise:

1. Does Sybil detection benefit more from longer DPs that will be broadcast less frequently, or is it better to send shorter packets more often?

2. What is the DP-related overhead (medium blocking) for different τ and T to achieve a required reliability?

The answer to the first question can be found in Figure 5.5, which shows multiple graphs, each of which represents the probability of not detecting a delay for specific DP durations and periods. The values for τ were the same as used in Figure 5.4, and the values for q were their corresponding average value. The notation T_{100}, T_{500} and T_{1000} was used to denote periods of T = 100, 500 and 1000ms respectively. The x-axis considers the algorithm's execution time in seconds, and the y-axis the probability of non-detection of an attack. The plots are stepping functions, as ϵ decreases with each additional DP. As expected from Equation 5.3, higher detection probabilities are

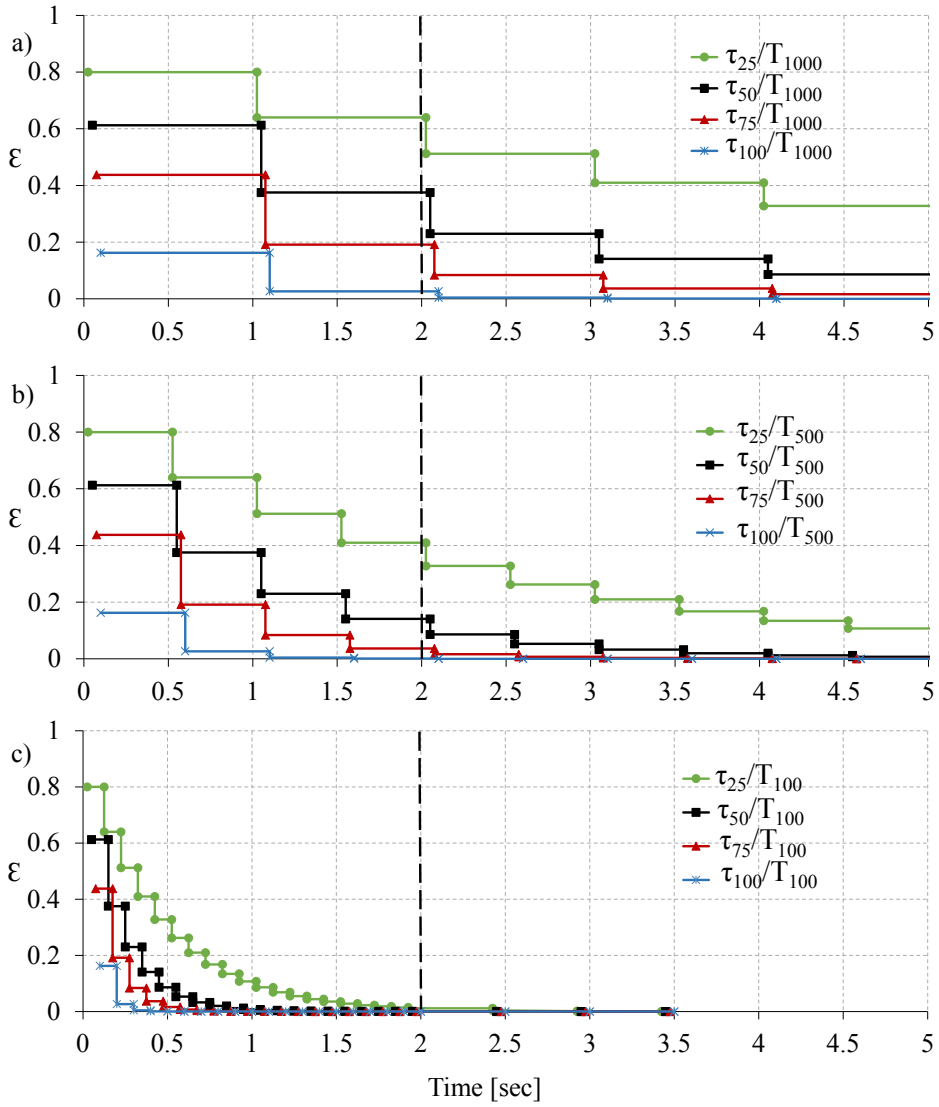


FIGURE 5.5: Probability of not detecting a delay for DPs with different τ and T

achieved for all τ as more DPs are considered, i.e. as N increases. The objective of the Sybil detection algorithm should be to have a reasonably high detection probability in the shortest time. As is obvious from comparing the graphs in Figure 5.5a), b) and c), smaller T result in shorter times to improve ϵ. This however comes at the cost of higher medium blocking.

Let's consider the detection interval $T_{detect} = 2s$, as in the discussion above. This cutoff time is indicated by a vertical dashed line in Figure 5.5. For a given time, period T dictates how many DPs are sent. In our case of 2 seconds, T_{100} results in 20 DPs, T_{500} in 4, and T_{1000} in 2 DPs. For long T this poses a problem for achieving

a low ϵ, as there are only few DPs contributing to Equation 5.3. For example, in the case of T_{1000}, the detection algorithm has only two chances to detect a delay. In the case of Figure 5.5a) with T_{1000}, only τ_{100} was able to achieve a low uncertainty, below 0.03 in plot τ_{100}/T_{1000}, thus achieving a reliability of greater than 0.97. All other τ resulted in unacceptable ϵ, and thus unacceptable reliabilities.

Question 2 addressed the overhead, i.e., medium blocking, for detecting delays. Figure 5.6 shows the DP overhead normalized over the detection interval T_{detect}. This

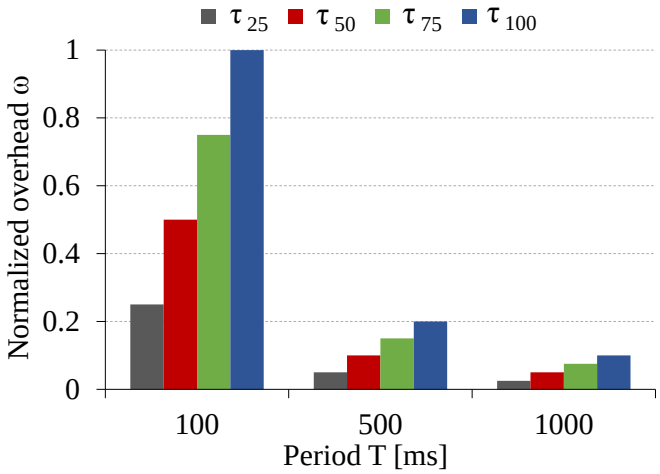


FIGURE 5.6: Overhead of DPs with different τ and T

ratio for medium blocking is denoted by ω. Obviously, overhead ratio ω for longer periods, e.g., T_{1000}, is lower than that of shorter periods, e.g., T_{100}, for any given duration τ. The reason for this is that for longer T fewer DPs are sent during T_{detect} than for shorter T. Furthermore, for a given T, longer durations τ result in higher overhead ratios than short τ. Overhead was calculated based on the assumption that one sends DPs during the entire interval T_{detect}. This however is not necessary, as one only needs to run detection until a specified ϵ is reached.

Figure 5.7 shows the overhead ratio ω associated with DP during the 2 seconds to achieve 90% detection probability, i.e., $\epsilon = 0.1$. Recall that Algorithm 1 terminates once a specified ϵ has been reached. It should be noted that a zero entry in the graph indicates that the specified ϵ could not be achieved in the given time. For T_{500} only $\tau = 75$ and 100 resulted in detection, whereas for T_{1000} only $\tau = 100$ can achieve the specified detection probability. To answer question 2) above, from a medium

blocking point of view, it is best to select the largest T with the τ producing the smallest ω. Preference is given to the largest T, in order to spread the DPs as wide as possible, thereby minimizing monopolization of the medium by DPs and allowing BSMs, including those from other vehicles, to be transmitted.

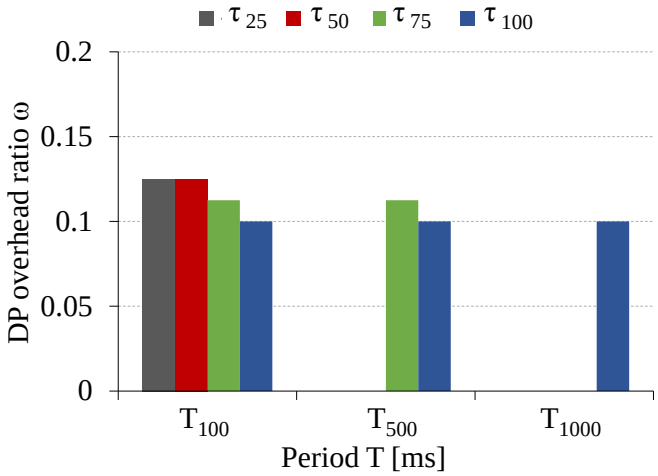


FIGURE 5.7: Overhead Ratio ω to achieve $\epsilon = 0.1$ during 2s

5.5 CONCLUSION

In this chapter we investigated Sybil attack detection in VANETs operating in dynamic power environments. An enhanced active Sybil detection algorithm capable of mitigating against Sybil attacks in such complicated environment was presented. This new algorithm is based on detecting queuing delays of nodes subjected to Detection Packets to determine if a BSM is valid or has been spoofed. In order to minimize the impact of these packets on medium usage delay detection should be spread over time. Analysis and field experiments on the impact of periods T and durations τ of detection packets on Sybil node detection showed that it is best to use the largest periods with the lowest overhead ratio ω that can achieve a predefined detection probability.

CHAPTER 6

Conclusions and Future Work

6.1 CONCLUSIONS

The main focus of this research was security and reliability of DSRC safety applications operating in vehicular networks subjected to malicious act. Specifically, we considered wireless hybrid jamming and Sybil attacks as the source of attacks. The critical nature of DSRC safety applications, with their severe consequences in case of failure, requires high reliability against potential failures. Different security mechanisms such as digital signatures, encryption and authentication have been implemented in the current standards, however, these mechanisms are not sufficient to mitigate against these types of attacks.

This research proposed different algorithms and strategies to improve reliability of safety applications operating in hostile environments. Specifically, four algorithms were proposed, which can help to enhance resilience of safety applications against the impact of maliciously induced faults. The feasibility of these approaches was tested through simulations and field experiments. The results were analyzed to show their effectiveness.

First, a new hybrid jammer was presented together with an effective detection algorithm. This jammer exposed queuing behavior that can be exploited by attackers to cause safety application failure. It can make innocent nodes appear misbehaving by making them look to excessively use the medium. The field tests results showed that the new detection algorithm can not only detect the presence of a hybrid jammer, but can also distinguish between misbehaving nodes and other nodes that are impacted by the jammer. Current PDR-based detection methods fail to detect the new jammer.

Second, a new Enhanced Voting-based Algorithm (EVA) was presented to improve the reliability of DSRC safety applications subjected to event fabrication by malicious entities or suppression of real event notifications. The algorithm operates

in two stages. The first stage uses specific metrics to discover misbehaving nodes. The second stage considers delayed BSMs messages coming from victim nodes, which in turn directly affect decisions of safety applications. Field experiments were conducted, and the results observed showed that the EVA was capable of significantly reducing the safety application's decision times, thereby improving their reliability. In the worst case attack scenarios, the improvements of the EVA were significant. During experiments enhancements of up to 3.3 seconds were observed. In the context of safety critical applications, such improvements could have significant impact on avoiding accidents and saving lives.

Third, a passive algorithm to detect Sybil attacks for static power environments was proposed, which can cause serious problems for DSRC safety applications that using voting schemes. The passive algorithm runs in two phases. In phase 1 it checks a power field in the BSM header to catch Sybil vehicles that might manipulate the transmission power. In the next phase and algorithm that changes Rx sensitivity is applied. Based on the selected sensitivity the number of vehicles expected in the reception area is calculated. Any deviation of this number from the actual number of vehicles observed might indicate a Sybil attack. Results show that the passive algorithm is only suitable for static power environments, but ineffective for dynamic power environments, which are much more challenging. Thus, an active algorithm was introduced, which is suitable for Sybil detection in dynamic power environments. This algorithm can locate Sybil nodes using short Detection Packets (DPs) without adding special hardware or information exchanges. This will help safety application to reject values from Sybil nodes, which is of great importance for solutions based on voting schemes. Analysis and field experiments on the effects of the number of BSMs, as well as the DP duration and time placement accuracy, revealed that the placement and duration of these packets have the largest impact on Sybil detection effectiveness.

Lastly, an enhanced active Sybil detection algorithm was presented, which is capable of mitigating against Sybil attacks in dynamic power environments. This enhanced algorithm is based on detecting queuing delays of nodes subjected to DPs to determine if a BSM is valid or has been spoofed. In order to minimize the impact

of DPs on medium usage, delay detection should be spread over time. Analysis and field experiments on the impact of DP periods and durations on Sybil node detection probabilities showed that it is best to use larger DP periods with the lowest overhead ratio that can achieve a predefined detection accuracy.

6.2 FUTURE WORK

The research presented was focused on DSRC communication. It could be of great benefit to study the impact of multi-sensor technologies on the effectiveness of the solutions presented. This may be of special interest to autonomous vehicles, as they are already equipped with diverse technologies such as radar, lidar and cameras. On the other hand, DSRC capability will extend the effectiveness of these sensor technologies beyond the line of sight.

The safety applications considered in the research were aiming at reducing the probability of rear end collisions. Applying the solutions presented to other DSRC safety applications may expose different attack challenges and defense opportunities. The different assumptions related to speeds, distances and directions could expose specific problematics.

Finally, the impact of traffic density on the limits and scalability of solutions should be studied. Specifically, the trade-off space between communication range, vehicle-density-induced packet utilization of the medium, and message length as it results from different data rates, should be analyzed.

Bibliography

[1] *Traffic Safety Facts: Crash Stats*, U.S. Department of Transportation, National Highway Traffic Safety Administration, DOT HS 812 219, Nov 2016.

[2] General Motors, *https://www.gm.com/all-news-stories/technology.html*

[3] J. B. Kenney, *Dedicated Short-Range Communications (DSRC) Standards in the United States*, Proceedings of the IEEE, vol. 99, no. 7, pp. 1162-1182, 2011.

[4] *Amendment of the Commission's Rules Regarding Dedicated Short-Range Communication Services in the 5.850-5.925 GHz Band (5.9 GHz Band)*, Federal Communications Commission FCC 03-324, 2004.

[5] *Vehicle safety communications-applications (VSC-A) final report*, DOT HS 811 492 A. U.S. Department of Transportation, NHTSA, September 2011.

[6] *Standard Specification for Telecommunications and Information Exchange Between Roadside and Vehicle Systems - 5 GHz Band Dedicated Short Range Communications (DSRC) Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, ASTM E2213-03, 2010.

[7] *IEEE Standard for Information technology–Telecommunications and information exchange between systems–Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments*, IEEE Std 802.11p, 2010.

[8] *IEEE Draft Guide for Wireless Access in Vehicular Environments (WAVE) -Architecture*, IEEE P1609.0, September 2012.

[9] *IEEE Standard for Wireless Access in Vehicular Environments - Security Services for Applications and Management Messages*, IEEE Std 1609.2TM, 2013.

[10] *IEEE Standard for Wireless Access in Vehicular Environments (WAVE) - Networking Services*, IEEE Std 1609.3TM, 2010.

[11] *IEEE Standard for Wireless Access in Vehicular Environments (WAVE) - Multi-Channel Operation*, IEEE Std 1609.4TM, 2010.

[12] *Dedicated Short Range Communications (DSRC) Message Set Dictionary. Society of Automotive Engineers*, SAE J2735, November 2009.

[13] Andreas. F. Molisch, *Wireless Communications*, 2nd edition, John Wiley & Sons Ltd, 2014, ISBN: 978-0-470-74187-0.

[14] *IEEE Standard for Information technology–Telecommunications and information exchange between systems–Local and metropolitan area networks - Specific requirements, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements*, IEEE Std 802.11e, 2005.

[15] Luuk Hendriks, *Effects of Transmission Queue Size, Buffer and Scheduling Mechanisms on the IEEE 802.11p Beaconing Performance*, 5th Twente Student Conference on IT, June 20, Enschede, NL, 2011.

[16] Vishal Kumar, Shailendra Mishra, Narottam Chand, *Applications of VANETs: Present & Future* Communications and Network, vol. 5, pp. 12-15, 2013.

[17] M. Faezipour, M. Nourani, A. Saeed and S. Addepalli, *Progress and Challenges in Intelligent Vehicle Area Networks* Communication of the ACM, vol. 55, no. 2, pp. 90-100, 2012.

[18] *Vehicle Safety Communications Project Final Report: Identify Intelligent Vehicle Safety Applications Enabled by DSRC*, U.S. Department of Transportation, National Highway Traffic Safety Administration, 2005.

[19] *Pre-Crash Scenario Typology for Crash Avoidance Research*, National Highway Traffic Safety Administration, DOT HS 810 767, April, 2007.

[20] Arada Systems, *www.aradasystems.com*

[21] Aravendra K. Sharma, Sushil K. Saroj, Sanjeev K. Chauhan and Sachin K. Saini, *Sybil attack prevention and detection in vehicular ad hoc network*, IEEE International Conference on Computing, Communication and Automation (ICCCA), 2016

[22] M. Al-Mutaz, L. Malott, S. Chellappan, *Detecting Sybil Attacks in Vehicular networks*, Journal of Trust Management, Springer, pp. 2-19, 2014.

[23] K. Pelechrinis, M. Iliofotou and S. V. Krishnamurthy, *Denial of Service Attacks in Wireless Networks: The Case of Jammers*, in IEEE Communications Surveys & Tutorials,, vol.13, no.2, pp.245,257, 2^{nd} Quarter 2011.

[24] Mohamed S. Mohamed, P. Dandekhya and Axel Krings, *Beyond Passive Detection of Sybil Attacks in VANET*, The 6th IEEE International Conference on Reliability, Infocom Technologies and Optimization (ICRITO), 2017.

[25] M. Raya, and J. Hubaux, *The Security of Vehicular Ad Hoc Networks,* in Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks. ACM, 2005.

[26] Xu W, Trappe W, Zhang Y, Wood T, *The feasibility of launching and detecting jamming attacks in wireless networks* In: Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp 46-57, 2005.

[27] M. Li, S. Salinas, P. Li, J. Sun, and X. Huang, *MAC-Layer Selfish Misbehavior in IEEE 802.11 Ad Hoc Networks: Detection and Defense*, IEEE TRANSACTIONS ON MOBILE COMPUTING, VOL. 14, NO. 6, JUNE 2015.

[28] P. Kyasanur and N. Vaidya, *Selfish MAC layer misbehavior in wireless network*, IEEE Trans. on Mobile Computing, vol. 4, no. 5, pp. 502-516, Sep. 2005.

[29] Z. Lu, W. Wang, and C. Wang, *On order gain of backoff misbehaving nodes in CSMA/CA-based wireless networks*, in Proc. IEEE Conf. Comput. Commun., San Diego, CA, USA, Mar. 2010.

[30] L. Toledo, and X. Wang, *Robust detection of MAC layer denial-of-service attacks in CSMA/CA wireless networks*, IEEE Trans. Inf. Forensics Security, vol. 3, no. 3, pp. 347-358, Sep. 2008.

[31] M. Raya, J. Hubaux, and I. Aad, *Domino: A system to detect greedy behavior in ieee 802.11 hotspots*, in Proc. ACM 2nd Int. Conf. Mobile Syst., Appl. Serv., Boston, MA, USA, Jun. 2004.

[32] M. N. Mejri and J. Ben-Othman, *Detecting greedy behavior by linear regression and watchdog in vehicular ad hoc networks*, IEEE Global Communications Conference (GLOBECOM), pp.5032 -5037, 2014.

[33] M. N. Mejri and J. Ben-Othman, *Entropy as a new metric for denial of service attack detection in vehicular ad-hoc networks*, In Proceedings of the 17th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems, 2014.

[34] Vinh Hoa LA, Ana Cavalli, *Security Attacks and Solutions in Vehicular Ad Hoc Networks: A Survey*, International Journal on AdHoc Networking Systems (IJANS) Vol. 4, No. 2, April 2014.

[35] J. Newsome, E. Shi, D. Song, and A. Perrig, *The Sybil Attack in Sensor Networks: Analysis & Defenses*, In International symposium on information processing in sensor networks, pages 259?268, 2004.

[36] J. Douceur *The Sybil Attack*, In First International Workshop on Peer-to-Peer Systems, pages 251?260, March 2002.

[37] C. Piro, C. Shields, B. N. Levine, *Detecting the Sybil attack in mobile ad hoc network*, in proceedings of the International Conference on Security and Privacy in Communication Networks, pp. 1?11, 2006.

[38] J. Petit and Z. Mammeri, *Dynamic consensus for secured vehicular ad hoc networks*, in Wireless and Mobile Computing, Networking and Communications (WiMob), IEEE 7th International Conference on. IEEE, 2011, pp. 18, 2011.

[39] Hani Alturkostani and Axel Krings, *The Impact of Jamming on Threshold-Based Agreement in VANET*, International Conference on Connected Vehicles and Expo (ICCVE), 2014.

[40] Z. Cao, J. Kong, U. Lee, M. Gerla, and Z. Chen, *Proof-of-relevance: Filtering false data via authentic consensus in vehicle ad-hoc networks*, in INFOCOM Workshops, IEEE. IEEE, 2008, pp. 16, 2008.

[41] Yu-Chih Wei and Yi-Ming Chen, *Adaptive Decision Making for Improving Trust Establishment in VANET*, IEICE - Asia-Pacific Network Operation and Management Symposium (APNOMS), 2014.

[42] W. B. Johnson, *Design and Analysis of Fault-Tolerant Digital Systems*, Addison-Wesley Publishing Company, New York, 1989.

[43] A. Serageldin, H. Alturkostani and A. Krings, *On the Reliability of DSRC Safety Applications: A Case of Jamming*, in IEEE International Conference on Connected Vehicles and Expo (ICCVE), Las Vegas, NV, pp. 501-506, 2013.

[44] B. Ostermaier, F. Dotzer, and M.Strassberger, *Enhancing the security of local danger warnings in vanets - a simulative analysis of voting schemes*, in Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on. IEEE, pp. 422431, 2007.

[45] X. Ma, X. Yin, and K.S. Trivedi, *On the Reliability of Safety Applications in VANETs*, Invited paper, International Journal of Performability Engineering Special Issue on Dependability of Wireless Systems and Networks, 8(2), March 2012.

[46] M. Demirbas and Y. Song, *An RSSI-based scheme for Sybil attack detection in wireless sensor networks*, Proc. IEEE Int. Symp. on a World of Wireless Mobile and Multimedia Networks, 2006.

[47] B. Xiao, B. Yu, and C. Gao, *Detection and localization of Sybil nodes in VANETs*, Proceedings of the Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS '06), Los Angeles, CA, USA, pp. 1-8, 2006.

[48] Mohamed S. Bouassida, et. al., *Sybil Nodes Detection Based on Received Signal Strength Variations within VANET*, International Journal of Network Security, Vol.9, No.1, PP.22-33, 2009.

[49] Caffery J, Stuer GL., *Overview of radio location in CDMA cellular systems*, IEEE Transactions on Vehicular Technology, 36(4):38-45, Apr 1998.

[50] Alam MS.,Alsharif S.,Haq N., *Efficient CDMA wireless,position,location system using TDOA method*, International Journal of Communication Systems, 24(9):1230?1242, Sep. 2011.

[51] S. Hussein, Axel Krings, and Azad Azadmanesh, *VANET Clock Synchronization for Resilient DSRC Safety Applications*, in Proc. 7th Resilience Week Symposia, Wilmington, DE, (7 pages), September 18-22, 2017.

[52] Jyoti Grover, Manoj Singh Gaur and Vijay Laxmi, *A Sybil Attack Detection Approach using Neighboring Vehicles in VANET*, Proc. of the 4th Int. conference on Security of information and networks, 2011.

[53] Kamran Zaidi, Milos Milojevic, Veselin Rakocevic, Arumugum Nallanathan and Muttukrishnan Rajarajan, *Host Based Intrusion Detection for VANETs: A Statistical Approach to Rogue Node Detection*, IEEE Transactions on Vehicular Technology, 2015.

[54] Chen Chen, Xin Wang, Weili Han, and Binyu Zang, *A Robust Detection of the Sybil Attack in Urban VANETs*, 29th IEEE International Conference on Distributed Computing Systems Workshops, 2009.

[55] Soyoung Park, et. al., *Defense against Sybil attack in vehicular ad hoc network based on roadside unit support*, MILCOM, pp. 1-7, 2009.

[56] A. Khalili, J. Katz, and W. Arbaugh, *Toward secure key distribution in truly ad-hoc networks*, in Proceedings of IEEE Workshop on Security and Assurance in Ad hoc Networks, 2003.

[57] M. Raya and J.P. Hubaux, *Securing vehicular ad hoc networks*,Journal of Computer Security, 15(1), 39-68, 2007.

[58] M. Rahbari, M. Ali and J. Jamali, *Efficient detection of Sybil attack based on cryptography in VANET*, International Journal of Network Security and Its Applications IJNSA, Vol.3, No.6, November 2011.

[59] D. M. Pozar, *Microwave Engineering 3rd Edition*, NY, Wiley, 2010.

[60] NS3 Simulation, *https://www.nsnam.org/docs/tutorial/html/index.html*

[61] S. Hussein, M. S. Mohamed and Axel Krings, *A New Hybrid Jammer and its Impact on DSRC Safety Application Reliability*, The 7th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2016.

[62] A. Avizienis, J. C. Laprie, B. Randell and C. Landwehr, *Basic concepts and taxonomy of dependable and secure computing*, in IEEE Transactions on Dependable and Secure Computing, vol. 1, no. 1, pp. 11-33, Jan.-March 2004.

[63] D. Dolev et. al., *Reaching Approximate Agreement in the Presence of Faults,* JACM, Vol. 33, No. 3, pp. 499 - 516, July 1986.

[64] D. Dolev, N.A. Lynch, S.S. Pinter, E.W. Stark, and W.E. Weihl, *Reaching Approximate Agreement in the Presence of Faults,* Proc. of the Third Symp. Reliability in Distributed Software and Database Systems, pp. 145 - 154, Oct. 1983.

[65] F. J. Meyer and D. K. Pradhan, *Consensus with Dual Failure Modes,* in IEEE Transactions on Parallel and Distributed Systems, vol. 2, no. 2, pp. 214-222, Apr 1991.

[66] M. Pease, R. Shostak and L. Lamport, *Reaching Agreement in the Presence of Faults,* in Journal of the ACM (JACM), 1;27(2):228-34, Apr 1980.

[67] L. Lamport, R. Shostak and M. Pease, *The Byzantine Generals Problem,* in ACM Transactions on Programming Languages and Systems (TOPLAS), 1;4(3):382-401, Jul 1982.

[68] P. Thambidurai and Y.K. Park, *Interactive consistency with multiple failure modes*, in Proceedings of the Seventh Symposium on Reliable Distributed Systems, Columbus, OH, pp. 93-100, 1988.

[69] K. Sjoberg, E. Uhlemann and E. G. Strom,*How Severe Is the Hidden Terminal Problem in VANETs When Using CSMA and STDMA*, IEEE Vehicular Technology Conference (VTC Fall), San Francisco, 2011

[70] Kyoungsoo Bok, Jongtae Lim, Seungwan Hong, Jaesoo Yoo, *A multiple RSU collaborative scheduling scheme for data services in vehicular ad hoc networks*, Journal of Cluster Computing, Volume 20 Issue 2, Pages 1167-1178, June 2017.

[71] M. H. Azadmanesh and R. M. Kieckhafer, *Exploiting omissive faults in synchronous approximate agreement*, IEEE Transactions on Computers, vol.49, no.10, pp.1031,1042, Oct 2000.

[72] Paul J. Weber, *Dynamic Reduction Algorithms for Fault Tolerant Convergent Voting with Hybrid Faults*, Order No. 3209907, Michigan Technological University, Ann Arbor, 2006.

APPENDIX A

NS-3 Simulation Functions

The following are the NS-3 Simulation functions that have been used for *Passive Sybil Detection Algorithm* in Chapter 4. The code has been divided into several functions, in order to generate mobility models, propagation models, and change sensitivity. The simulation functions were used to simulate the attack scenarios depicted in Figure 4.4.

A.1 STATIONARY POSITION MODEL

```
//Using ConstantPositionMobility Model

//Specify two nodes to be stationary at specific positions
 MobilityHelper mobility;
 Ptr<ListPositionAllocator> positionAllocator = CreateObject<
    ListPositionAllocator>();
 positionAllocator->Add(Vector(0, 150, 0));
 positionAllocator->Add(Vector(100, 150, 0));
 mobility.SetPositionAllocator(positionAllocator);
 mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel
    ");
 mobility.Install(bsmNodes);

//Specify nodes to be stationary but randomly allocated
 MobilityHelper mobility;
 mobility.SetPositionAllocator (
       "ns3::RandomDiscPositionAllocator",
       "Theta", RandomVariableValue(UniformVariable (0.0,
          6.2830)),
```

```
        "Rho", RandomVariableValue(UniformVariable (0.0, 50.0)),
        "X", DoubleValue (0.0),
        "Y", DoubleValue (0.0));
 mobility.SetMobilityModel ("ns3::
    ConstantPositionMobilityModel");
 mobility.Install(bsmNodes);
```

A.2 VELOCITY-MOBILITY MODEL

```
// Using ConstantSpeedMobility Model:

// Setup every node moving with the constant velocity during
   the simulation, i.e., no braking or acceleration.
// Each of the nodes are assigned with random velocity between
    15 and 25 m/s.
 MobilityHelper mobility;
 mobility.SetMobilityModel ("ns3::
    ConstantVelocityMobilityModel");
 mobility.Install (nodes);
 Ptr<UniformRandomVariable> rvar = CreateObject<
    UniformRandomVariable>();
 //Assigning Random velocity between (15m/s, 25m/s) in the x-
    direction for each nodes
 for (NodeContainer::Iterator i = nodes.Begin (); i != nodes.
    End (); ++i){
            Ptr<Node> node = (*i);
            double speed = rvar->GetValue(15, 25);
            node->GetObject<ConstantVelocityMobilityModel>()->
               SetVelocity(Vector(speed,0,0));
 }
```

A.3 PROPAGATION MODELS

```
//Configure Channel with FrissPropagationLossModel
YansWifiChannelHelper waveChannel;
waveChannel.AddPropagationLoss(
"ns3::FriisPropagationLossModel",
"Frequency", DoubleValue(5.860e9),
"SystemLoss", DoubleValue(1.0),
"MinLoss", DoubleValue(0.0));

//Configure Channel with LogDistancePropagationLossModel
YansWifiChannelHelper waveChannel;
waveChannel.AddPropagationLoss(
"ns3::LogDistancePropagationLossModel",
"Exponent", DoubleValue(3),
"ReferenceDistance", DoubleValue(1.0),
"ReferenceLoss", DoubleValue(46.6777));

//Configure Channel with ThreeLogDistancePropagationLossModel
YansWifiChannelHelper waveChannel;
waveChannel.AddPropagationLoss(
"ns3::ThreeLogDistancePropagationLossModel",
"Exponent0", DoubleValue(1.9),
"Distance0", DoubleValue(1.0),
"Exponent1", DoubleValue(3.8),
"Distance1", DoubleValue(200.0),
"Exponent2", DoubleValue(3.8),
"Distance2", DoubleValue(500.0),
"ReferenceLoss", DoubleValue(46.6777));
```

```
//Configure Channel with default PropagationLossModel (i.e.
   LogDistancePropagationLossModel)
YansWifiChannelHelper waveChannel = YansWifiChannelHelper::
   Default();
```

A.4 DETERMINING OF THE TRANSMISSION RANGE

```
//Code snippet from sybilattack-Tx_Range-Rx_Sensitivity.cc
         double rxSensitivity[] = { -97.0, -96.0, -95.0,
            -94.0, -93.0, -92.0, -91.0, -90.0, -89.0, -88.0,
            -87.0, -86.0, -85.0, -83.0, -82.0, -81.0, -80.0,
            -79.0, -78.0, -77.0, -76.0, -75.0, -74.0, -73.0,
            -72.0,-71.0, -70.0, -69.0, -68.0, -67.0, -66.0,
            -65.0, -63.0, -61.0, -59.0, -57.0, -55.0, -53.0,
            -50.0, -48.0, -45.0, -40.0, -35.0 };

        for (uint32_t i = 0; i < sizeof(rxSensitivity) /
           sizeof(rxSensitivity[0]);i++)
        {
                ss.str("");
                ss << "EnergyDetectionThreshold(" <<
                   rxSensitivity[i] << "dBm)";
                NS_LOG_DEBUG(ss.str());
                experiment = WaveExperiment(ss.str());
                experiment.SetEnergyDetectionThreshold(
                   rxSensitivity[i]);
                //Determine the max Transmission Range for the
                    Rx_Sensitivity
                experiment.Run();
                gnuplot.AddDataset(experiment.GetDataset());
```

```
            TxRangeDataset.Add(experiment.
                GetEnergyDetectionThreshold(),
                                experiment.
                                    GetMaxTransmissionRange());
            csvOutputStream << experiment.
                GetEnergyDetectionThreshold() << ","
                                << experiment.
                                    GetMaxTransmissionRange()
                                    << std::endl;
}
```

A.5 MAIN FUNCTION OF SIMULATION

```
/*
 * NS3-simulation.cc
 */

#include "ns3/command-line.h"
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/wifi-helper.h"
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/wifi-module.h"
#include "ns3/propagation-loss-model.h"
#include "ns3/mobility-module.h"
#include "ns3/netanim-module.h"
#include "ns3/position-allocator.h"
#include "ns3/udp-echo-helper.h"
#include "ns3/config-store.h"
#include "ns3/gtk-config-store.h"

#include <fstream>
#include <iostream>
#include <iomanip>

using namespace ns3;

NS_LOG_COMPONENT_DEFINE("SampleNS3Application");


int main (int argc, char *argv[])
```

```
{
uint32_t m_numNodes = 2;
double m_simTime = 10;
bool m_verbose = false;

LogComponentEnable("UdpEchoServerApplication",LOG_LEVEL_INFO);
LogComponentEnable("UdpEchoClientApplication",LOG_LEVEL_INFO);

/**** Set Default Attributes here ****/

/**** Parse Command Line Arguments here ****/
CommandLine cmd;
cmd.AddValue("numBsmNodes", "number of Nodes", m_numNodes);
cmd.AddValue("totalSimTime", "Total Simulation Time (seconds)"
    , m_simTime);
cmd.AddValue("verbose", "Turn on all Wifi log components",
    m_verbose);
cmd.Parse(argc, argv);

/**** Create the Network Topology ****/
/**** Create Nodes ****/
NS_LOG_INFO("Create Nodes");
NodeContainer nodes;
nodes.Create(m_numNodes);

/**** Configure Physical Layer ****/
NS_LOG_INFO("Configure Physical Layer");
YansWifiPhyHelper wifiPhy = YansWifiPhyHelper::Default();
// Override any default setting of the Physical layer if any
wifiPhy.Set("RxGain",DoubleValue(-10));
// ............
```

```
// . . . . . .
//Set the pcap file format (log file)
wifiPhy.SetPcapDataLinkType(YansWifiPhyHelper::DLT_IEEE802_11)
    ;

/**** Configure Channel ****/
NS_LOG_INFO("Configure␣Channel");
YansWifiChannelHelper wifiChannelHelper =
    YansWifiChannelHelper::Default();
// Override any default setting of the Channel if any
wifiChannelHelper.SetPropagationDelay("ns3::
    ConstantSpeedPropagationDelayModel");
wifiChannelHelper.AddPropagationLoss("ns3::
    LogDistancePropagationLossModel");
Ptr<YansWifiChannel> wifiChannel = wifiChannelHelper.Create();
// Associate the wifi channel with the wifi physical layer
wifiPhy.SetChannel (wifiChannel);


/**** Configure MAC Layer ****/
NS_LOG_INFO("Configure␣MAC␣Layer");
// Nqos refers to the MAC layer with no QOS(Quality of Service
    ).
// For QOS MAC, use QosWifiMacHelper
NqosWifiMacHelper wifiMac = NqosWifiMacHelper::Default();
// Set the MAC to either ApWifiMac, StaWifiMac, AdhocWifiMac
// based on how the node will behave in the network (as
    AccessPoint, Station or Adhoc Node)
wifiMac.SetType("ns3::AdhocWifiMac");
```

```
//WifiHelper help to put together the wifi NICs with phy layer
    , mac layer components on each nodes;
WifiHelper wifiHelper;
// Override any default setting of Mac Layer if any
wifiHelper.SetStandard (WIFI_PHY_STANDARD_80211b);
std::string phyMode="OfdmRate6MbpsBW10MHz"; //a predefined
    string constant that defines the modulation and data rate
//set the data rate for data and control messages
wifiHelper.SetRemoteStationManager ("ns3::
    ConstantRateWifiManager",
"DataMode", StringValue(phyMode),
"ControlMode",StringValue(phyMode));
if(m_verbose){
  wifiHelper.EnableLogComponents(); //Turn on all Wifi logging
}
NetDeviceContainer devices = wifiHelper. Install(wifiPhy,
    wifiMac, nodes); //devices is analogous to NIC cards in
    network devices


/**** Add (Internet) Stack in each nodes ****/
NS_LOG_INFO("Install␣Internet␣Stack");
InternetStackHelper stack;
stack.Install (nodes);

/**** Configure IP Addressing and Routing ****/
// Assign the IP address to the network interfaces
// Can also define Routing protocol to be used
Ipv4AddressHelper ipv4Address;
ipv4Address.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer interfaces;
```

```
interfaces = ipv4Address.Assign (devices);

/**** Configure Application in each Nodes ****/
NS_LOG_INFO("Configure␣Application");
// Install application on nodes.
  uint16_t port = 9;
  UdpEchoServerHelper server(port);
  ApplicationContainer apps = server.Install (nodes.Get (0));
  apps.Start (Seconds (1.0));
  apps.Stop (Seconds (m_simTime));

  uint32_t packetSize = 1024;
  uint32_t maxPacketCount = 1;
  Time interPacketInterval = Seconds (1.0);
  Address serverAddress = Address(interfaces.GetAddress(0));
  UdpEchoClientHelper client (serverAddress, port);
  client.SetAttribute ("MaxPackets", UintegerValue (
     maxPacketCount));
  client.SetAttribute ("Interval", TimeValue (
     interPacketInterval));
  client.SetAttribute ("PacketSize", UintegerValue (packetSize
     ));
  apps = client.Install (nodes.Get (0));
  apps.Start (Seconds (2.0));
  apps.Stop (Seconds(m_simTime));

/****Configure Mobility****/
NS_LOG_INFO("Configure␣Mobility␣Model.");
MobilityHelper mobility;
//place two nodes at specific positions
Ptr<ListPositionAllocator> positionAllocator = CreateObject<
```

```
    ListPositionAllocator>();
positionAllocator->Add(Vector(0, 150, 0));
positionAllocator->Add(Vector(100, 150, 0));
mobility.SetPositionAllocator(positionAllocator);
//Configure both BSM node to be stationary
mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel"
   );
mobility.Install(nodes);

/**** Save the current Configuration for the simulation ****/
// This step is optional but this is helpful to record the
   various configuration
// used in the simulation into the file for debugging
std::string m_outputAttributeFile = "OutputAttributes.xml";
Config::SetDefault("ns3::ConfigStore::Filename",
    StringValue(m_outputAttributeFile));
Config::SetDefault("ns3::ConfigStore::FileFormat", StringValue
   ("Xml"));
Config::SetDefault("ns3::ConfigStore::Mode", StringValue("Save
   "));
ConfigStore outputConfig;
outputConfig.ConfigureAttributes();

/****Run Simulation****/
NS_LOG_INFO("Simulation␣Started");
std::cout << "Simulation␣Started"<< std::endl;
Simulator::Stop(Seconds(m_simTime));
Simulator::Run();
Simulator::Destroy();
std::cout << "Completed␣Simulation"<< std::endl;
```

```
}
```

A.6 SYBIL ATTACK SCENARIO 1

```
//sybilattack-passive-detection-scenario1.cc
/*****************Configure Mobility
    **********************/
NS_LOG_INFO("Configure␣Mobility␣Model.");
double yPosition = 600.0;
double xPosition = 0.0;
double xMin = 10.0;
double xMax = 1200.0;

MobilityHelper mobility;

Ptr<ListPositionAllocator> positionAllocator =
   CreateObject<
                ListPositionAllocator>();

//Position the HV
positionAllocator->Add(Vector(0, yPosition, 0));

//Position the Honest Nodes
for (uint32_t i = 0; i < numRealNodes; i++) {
        xPosition = uv->GetValue(xMin, xMax);
        positionAllocator->Add(Vector(xPosition,
           yPosition, 0));
}
//Position the Sybil Nodes
double xMinSybil = 50;
double xMaxSybil = 900;
```

```
        for (uint32_t i = 0; i < numSybilNodes; i++) {
                xPosition = uv->GetValue(xMinSybil, xMaxSybil)
                    ;
                positionAllocator->Add(Vector(xPosition,
                    yPosition, 0));
        }

        //Position the Malicious Node
        xPosition = uv->GetValue(xMaxSybil, xMax);
        positionAllocator->Add(Vector(xPosition, yPosition, 0)
            );

        mobility.SetPositionAllocator(positionAllocator);
        mobility.SetMobilityModel("ns3::
            ConstantPositionMobilityModel");
        mobility.Install(bsmNodes);
```

A.7 SYBIL ATTACK SCENARIO 2

```
//sybilattack-passive-detection-scenario2.cc
        /********************Configure Mobility
            ****************/
        NS_LOG_INFO("Configure␣Mobility␣Model.");
        double yPosition = 600.0;
        double xPosition = 0.0;
        double xMin = 10.0;
        double xMax = 1200.0;

        MobilityHelper mobility;
```

```
Ptr<ListPositionAllocator> positionAllocator =
   CreateObject<
                ListPositionAllocator>();
//Position the HV
positionAllocator->Add(Vector(0, yPosition, 0));

//Position the Honest Nodes
for (uint32_t i = 0; i < numRealNodes; i++) {
        xPosition = uv->GetValue(xMin, xMax);
        positionAllocator->Add(Vector(xPosition,
           yPosition, 0));
}

//Position the Sybil Nodes
double xMinSybil = 500;
double xMaxSybil = 1200;
for (uint32_t i = 0; i < numSybilNodes; i++) {
        xPosition = uv->GetValue(xMinSybil, xMaxSybil)
           ;
        positionAllocator->Add(Vector(xPosition,
           yPosition, 0));
}

//Position the Malicious Node
xPosition = uv->GetValue(10.0, xMinSybil);

positionAllocator->Add(Vector(xPosition, yPosition, 0)
   );

mobility.SetPositionAllocator(positionAllocator);
```

```
mobility.SetMobilityModel("ns3::
    ConstantPositionMobilityModel");
mobility.Install(bsmNodes);
```