A Trust Management Framework for Vehicular Ad Hoc Networks

by

Rezvi Shahariar

Submitted for the Degree of Doctor of Philosophy

Supervised by Dr. Chris Phillips

School of Electronic Engineering and Computer Science
Queen Mary, University of London
United Kingdom

June 2023

Statement Of Originality

I, Rezvi Shahariar, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below, and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university. The copyright of this thesis rests with the author and no quotation from it or information derived from it may be published without the prior written consent of the author.

Signature:

Date:

Abstract

The inception of Vehicular Ad Hoc Networks (VANETs) provides an opportunity for road users and public infrastructure to share information that improves the operation of roads and the driver experience. However, such systems can be vulnerable to malicious external entities and legitimate users. Trust management is used to address attacks from legitimate users in accordance with a user's trust score. Trust models evaluate messages to assign rewards or punishments. This can be used to influence a driver's future behaviour or, in extremis, block the driver. With receiver-side schemes, various methods are used to evaluate trust including, reputation computation, neighbour recommendations, and storing historical information. However, they incur overhead and add a delay when deciding whether to accept or reject messages. In this thesis, we propose a novel Tamper-Proof Device (TPD) based trust framework for managing trust of multiple drivers at the sender side vehicle that updates trust, stores, and protects information from malicious tampering. The TPD also regulates, rewards, and punishes each specific driver, as required. Furthermore, the trust score determines the classes of message that a driver can access. Dissemination of feedback is only required when there is an attack (conflicting information). A Road-Side Unit (RSU) rules on a dispute, using either the sum of products of trust and feedback or official vehicle data if available. These "untrue attacks" are resolved by an RSU using collaboration, and then providing a fixed amount of reward and punishment, as appropriate. Repeated attacks are addressed by incremental punishments and potentially driver access-blocking when conditions are met. The lack of sophistication in this fixed RSU assessment scheme is then addressed by a novel fuzzy logic-based RSU approach. This determines a fairer level of reward and punishment based on the severity of incident, driver past behaviour, and RSU confidence. The fuzzy RSU controller assesses judgements in such a way as to encourage drivers to improve their behaviour. Although any driver can lie in any situation, we believe that trustworthy drivers are more likely to remain so, and vice versa. We capture this behaviour in a Markov chain model for the sender and reporter driver behaviours where a driver's truthfulness is influenced by their trust score and trust state. For each trust state, the driver's likelihood of lying or honesty is set by a probability distribution which is different for each state. This framework is analysed in Veins using various classes of vehicles under different traffic conditions. Results confirm that the framework operates effectively in the presence of untrue and inconsistent attacks. The correct functioning is confirmed with the system appropriately classifying incidents when clarifier vehicles send truthful feedback. The framework is also evaluated against a centralized reputation scheme and the results demonstrate that it outperforms the reputation approach in terms of reduced communication overhead and shorter response time. Next, we perform a set of experiments to evaluate the performance of the fuzzy assessment in Veins. The fuzzy and fixed RSU assessment schemes are compared, and the results show that the fuzzy scheme provides better overall driver behaviour. The Markov chain driver behaviour model is also examined when changing the initial trust score of all drivers.

Acknowledgements

First, I would like to thank my supervisor Dr. Chris Phillips. His guidance, advice, dedicated support, and encouragement helped me to accomplish this work. When I sought advice from him, he always replied instantly which included thoughtful ideas and suggestions to resolve the matter. These suggestions improve my knowledge and research-related skills which include but not limited to design, implement, and validate the framework. I believe I could not have a better supervisor than him for my PhD.

Next, I am also grateful to have Dr. John Schormans as my second supervisor and Dr. Michael Chai as my individual assessor for their valuable evaluations and comments on my research.

Moreover, I offer my thanks to the colleagues and friends at Queen Mary, University of London who made my time in London memorable.

Furthermore, I would like to thank my sponsor the Bangabandhu Overseas Scholarship from the University of Dhaka for their financial support over the past four years.

My special and earnest thanks go to my family for their gracious support and their everyday caring. They are the indispensable driving force for completing this work.

TO MY FAMILY

Table of Contents

List of Figures

List of Tables

Glossary

3DES	–	Triple DES
3ME	–	Three Malicious Event
AES	–	Advanced Encryption Standard
AHP	–	Analytical Hierarchy Process
ALRS	–	Adaptive Link-ability Recognition Scheme
AMLA	–	Authentication with Multiple Levels of Anonymity
ATA	–	Agent of Trusted Authority
ATMS	–	Adaptive Trust Management Scheme
BSM	–	Basic Safety Messages
BTEV	–	Blockchain-based Traffic Event and Trust Verification
CA	–	Central authority
CAM	–	Congestion Avoidance Message
CNN	–	Convolutional Neural Network
CRL	–	Certificate Revocation List
CS-DC	–	Compression Sensing Data Compression
DDoS	–	Distributed Denial of Services
DES	–	Data Encryption Standard
DIKE	–	Dynamic Key Management Scheme
DoS	–	Denial of Service
DOTM	–	Data-Oriented Trust Model
DPB	–	Driver Past Behaviour
DSRC	–	Dedicated Short-Range Protocol
DSS	–	Digital Signature Standard
DST	–	Dempster-Shafer Theorem
DTM	–	Distributed Trust Management
ECC	–	Elliptic Curve Cryptosystem
ECU	–	Electronic Control Unit
EDR	–	Event Data Recorder
EOTM	–	Entity-Oriented Trust Model
ERM	–	Event Reporting Message
EWM	–	Emergency Warning Message
GPS	–	Global Positioning System
GPSR	–	Greedy Perimeter Stateless Routing
HMAC	–	Hash Message Authentication Code
HTM	–	Hybrid Trust Model
IBOOS	–	ID-Based Online/Offline Signature
IBS	–	Identity-Based Cryptography
ID	–	Identify
IP	–	Internet Protocol
ITS	–	Intelligent Transport System
KGC	–	Key Generation Centre
LBS	–	Location-Based Services
MAC	–	Message Authentication Code
MVEDR	–	Motor Vehicle Event Data Recorder
NS	–	Network Simulator
OBU	–	On-Board Unit
PID	–	Pseudo-Identity
PKI	–	Public Key Infrastructure
PoE	–	Proof of Event
RGTE	–	Reputation-Based Global Trust Management
RMC	–	Reputation Management Center

RSA	–	Rivest–Shamir–Adleman
RSS	–	Received Signal Strength
RSU	–	Roadside Unit
RTA	–	Road Transport Authority
RTO	–	Regional Transport Office
SDN	–	Software Defined Networking
SMC	–	Secure Multiparty Computation
SI	–	Severity of Incident
TA	–	Trust Authority
TDRL-RP	–	Trust Based Deep Reinforcement Learning Framework
TPD	–	Tamper-Proof Device
TPM	–	Tamper-Proof Module
V2I	–	Vehicle-to-Infrastructure
V2V	–	Vehicle-to-Vehicle
VANETs	–	Vehicular Ad Hoc Networks
IPP	–	Incremental Punishment Policy
WSM	–	WAVE Short Message
VGA	–	VANET Grouping Algorithm
WAVE	–	Wireless Access in Vehicular Environment
XAI	–	Explainable AI

Chapter 1: Introduction

1.1 Introduction to VANETs

VANETs are a promising approach to providing traffic comfort, safety, and infotainment services to vehicles. A VANET has only vehicles equipped with an OnBoard Unit (OBU) and RoadSide Units (RSUs) as its main elements. There may be an authority or multiple levels of authority to monitor the activities of VANETs and to make vital decisions about vehicles and RSUs for example, to maintain normal operation in VANETs and to block access to services, as appropriate. In VANETs, RSUs are at fixed positions and installed alongside roads. RSUs can communicate with passing vehicles, other RSUs, and the authority. When vehicles communicate with other vehicles it is called Vehicle-to-Vehicle (V2V) communication, and when they communicate with the infrastructure it is called Vehicle-to-Infrastructure (V2I) or conversely, Infrastructure-to-Vehicle (I2V) communication when an infrastructure entity announces a message towards vehicles. Vehicles may use the Dedicated Short-Range Protocol (DSRC/IEEE 802.11p) to communicate with others in a VANET (Jiang et al [1]). The combination of IEEE 802.11p and IEEE 1609.x comprises the Wireless Access in Vehicular Environment (WAVE) for the VANET (Jiang et al [1]).

Now-a-days, VANETs are becoming a major part of the Intelligent Transport System (ITS). Researchers are devoting effort on VANETs and ITS research in order to improve communication among the vehicles and the infrastructure to reduce transport problems or to warn vehicles in advance. Each year many people die in road accidents. From 2015 to 2019, at least 1730 people died due to road accidents each year across Great Britain. During the pandemic this number reduced but has subsequently risen as activity returns to normal (https://www.gov.uk/government/collections/road-accidents-and-safety-statistics [2]). As of September 2022, there were 40.8 million vehicles using the roads in the UK (https://www.racfoundation.org/motoring-faqs/mobility#a1 [3]). Over the past two years, the number of vehicles increased by 3.1%. In future, we can speculate that even more vehicles will use the roads as the trend is rising. In Europe, there are approximately 285 million vehicles in 2020 which has increased by 9.6% over the last five years (https://www.acea.be/statistics/article/report-vehicles-in-use-europe-2020/ [4]). In the USA, this number was 286 million in the first quarter of 2023(https://www.statista.com/statistics/859950/vehicles-in-operation-by-quarter-united-states/ [5]). Thus, we need a better traffic management system to manage the events on roads for getting timely updates via trustworthy and accurate message communication among vehicles and the infrastructure. This communication is required to reduce the aftereffects or consequences of hazardous situations which occur on roads or to improve traffic comfort for the drivers linked to planned route maintenance. According to the DSRC standard, vehicles can exchange situation awareness messages, Congestion Avoidance Messages (CAMs) and the Basic Safety Messages (BSMs) as warning messages for avoiding collisions or anticipated traffic issues.

Typically, a vehicle may share emergency information concerning the status of a road among vehicles and RSUs or they may request a service from the infrastructure, for example, about the location of a nearby petrol pump or parking area. VANETs can also support infotainment applications, e-tolling, and Internet communications to their users. VANETs can be deployed to mitigate the consequences of road incidents and to warn vehicles in advance. However, as this application involves frequent communication in an open medium, the application is at risk from security attacks in regard to the messages and infrastructure. For example, VANETs may suffer from: denial of service, untrue message, sybil, message alteration, black hole, and replay attacks (Lu et al [6]). Additionally, users can fraudulently announce false messages. Due to the nature of this application, messages in VANETs must be accurate and trustworthy. Otherwise, with an untrue mal-intent message, a vehicle can mislead many other vehicles causing congestion or other undesirable phenomena. Suppose a wrongdoer wants his path to be unobstructed; he could simply send fake messages containing congestion information to other vehicles to cause them to detour. As a result, drivers who believe the untrue message, help the wrongdoer in achieving his goal. In this way, mischievous drivers can diminish the traffic safety and comfort driving in VANETs. As another example, suppose a vehicle "V" announces a message reporting a crash as shown in Fig. 1-1. This message could be truthful or false. If other vehicles receive a false message, their subsequent detour will impact on their travel time as they believe the message. This will reduce the traffic comfort of affected drivers who take the detour. However, for a truthful announcement, the detour permits them to avoid potential congestion.

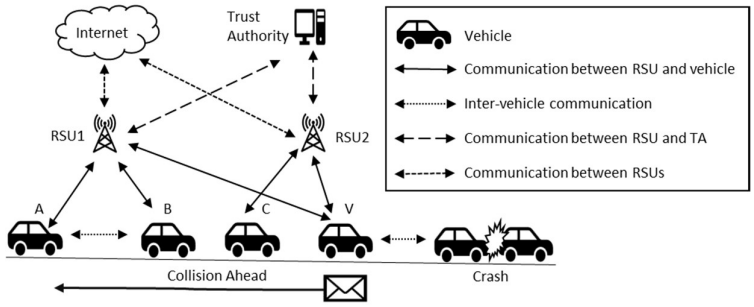


Fig. 1-1. Typical Mechanism for Reporting an Accident

In a VANET, outsider attacks or attacks from unauthorized users can be thwarted using a cryptographic scheme but not insider attacks. Cryptographic schemes i.e., hashing, digital signatures, encryption and decryption methods, and certificates are used for thwarting outsider attacks. Trust-based approaches are used to limit insider attacks from malicious authorized users (Tangade et al [7], Wei et al [8], and Tangade et al [9]). It is noted in (Dahiya et al [10] and Ahmad et al [11]) that trust schemes can improve the security by identifying dishonest vehicles and revoking messages from them. A trust management framework supports security services including access control, authentication, mischievous vehicle isolation and punishment (Gazdar et al [12]). Even so, trust approaches cannot protect VANETs completely (Tangade et al [7]). Basically, the trust that vehicle W attributes to vehicle

V is the confidence W places in a set of actions from V. It is not guaranteed that a trusted vehicle will always broadcast trustworthy messages. Hence, an evaluation is necessary either to reward or to punish a source vehicle and treat them accordingly. Typically, the reliability of relayed information is periodically evaluated using predefined metrics and computational methods (Gazdar et al [12]). Vehicles which consistently maintain a good trust score can be considered trustworthy by others as their current trust score implies that they announced trustworthy messages earlier. Thus, by evaluating announcements, trust management can play a major role in protecting the VANET from mischievous actions as they help to identify the malicious vehicles from their actions.

1.2 Motivation and Objectives

In existing approaches (Wei et al [8], Tangade et al [9], Gazdar et al [12], Haddadou et al [13], and Mühlbauer et al [14]) both trusted and untrusted vehicles can broadcast messages. Untrusted vehicles are expected to broadcast more malicious messages than trusted vehicles, which produces an additional demand on the network both in terms of message volume and the verification process. This places a considerable burden on the receivers. Approaches should aim to minimise the performance cost and communication overhead for VANETs. Even so, many approaches cross-check the event location, data freshness, and the role or experience of the source vehicle. Methodologies based on direct and indirect trust require regular monitoring of activities across both single and multi-hop transmission ranges. Some approaches (Wei et al [8], Dahiya et al [10], and Guleng et al [15]) result in excessive trust metric dissemination to verify the original announcement. These messages along with the event announcement complicate the overall situation as it is necessary to evaluate the validity of events quickly due to fast vehicle movement (Lu et al [6]). The authors in Gazdar et al [12] claim receivers should decide the trust of message in a short timeframe. However, when receivers independently compute trust from their neighbour's trust metrics, they suffer from a high response time as well (Wei et al [8], Gazdar et al [12], and Guleng et al [15]). Alternatively, approaches which allow trust computation at a centralized server need to communicate to obtain updated trust information concerning the source vehicle. This introduces additional delay in the decision-making process regarding emergency events. Consequently, some vehicles may drive into the event zone despite being warned, as suggested in Wei et al [16]. Also, there is an open debate (Dahiya et al [10] and Mühlbauer et al [14]) regarding how often a centralized server should communicate for revised trust data. We believe this problem can be addressed with the introduction of a sender-side trust management framework through access control and service association with a vehicle's trust score in the VANET.

This is based on a simple straightforward trust computational method which is more desirable when performance and communication efficiency are necessary. To this end, this research has rigorously surveyed existing trust models for VANETs. It has identified that many of these approaches suffer from performance and complexity issues, and considerable communication overhead. Lacking an efficient

trust management framework is the stimulus towards developing a novel trust management framework which can enhance the overall security of VANETs. To the best of my knowledge, existing approaches do not consider trust while announcing messages from the sender side. Also, many approaches use complex distributed methodologies for trust computation at the receiving vehicle or at a reputation server or an RSU. Receiver-side trust management frameworks allow the generation of messages from vehicles without first checking the trust. They compute the trust of sender messages after their arrival. If they discover the trust of sender vehicle and/or its message is lower than a threshold, then they ignore the message, otherwise they accept the message. To the best of my knowledge, none of the existing approaches consider a trust-based access control mechanism. As a result, both trusted and untrusted vehicles can announce messages. Thus, my research aims to create an efficient and effective mechanism for managing trust in VANETs. The research objective can be stated as: "How to regulate announcements based on the trust score of drivers from the sending side and to reduce overhead and response time as well as to enrich security for VANETs?"

This research overcomes the limitations of existing approaches by proposing a trust computation and management framework incorporating a sender-side Tamper-Proof Device (TPD) to ensure trustworthy message dissemination with the notion of access control. When sender-side trust evaluation is employed, there is no need to communicate further with neighbours to verify trust using trust metrics / feedback. This reduces communication overhead as there is no need to generate additional approach-inherent messages. Drivers can immediately believe the announcement as it is regulated by the TPD of the sender vehicle which results in almost zero driver decision time / response time. Here, we ignore processing time at the receiver end since in these days computing speed has greatly improved so whenever a module receives a message it can act on it almost instantly (as modelled in our simulator). It is seen in approaches such as Mühlbauer et al [14], that receivers wait up to a timer deadline to collect messages about an event. After that receivers evaluate the event to issue feedback about the event to the RSU which is required to update the reputation of sender vehicle. In this way, all receivers verify an event which induces high communication overhead and response time. Another approach proposed by Wei et al [8] uses Dempster-Shafer Theory (DST) to determine recommendation trust. Alternatively, Guleng et al [15] consider sending hello messages periodically among neighbours and then computing fuzzy logic-based direct trust and employing Q learning to compute indirect trust. With sender-side evaluation, there is no need to collect indirect trust metrics from neighbours. It removes false recommendations based on incorrect trust computation from the indirect neighbours.

This approach also relieves all receivers from the burden of trust computation of the sender vehicle and its messages. This framework assumes a TPD is equipped within every regular vehicle which manages trust and protects the trust score from unauthorized manipulation. The underlying principle of this framework is the implementation of access control when a driver is attempting to announce events. Another important feature of this framework is that it incorporates an untrue attack detection scheme

which runs only at the RSU whenever it receives conflicting information from multiple drivers. With this scheme, an RSU can detect untrue attacks and inconsistent attacks and punish vehicles appropriately when malicious behaviours are proven. Malicious drivers receive punishment from the RSU, and, in some cases, malicious drivers of vehicles are blacklisted when they broadcast untrue messages thrice. The obvious advantage of this scheme is that the receiver vehicles do not need any further communication with neighbours to evaluate the trust of a sender vehicle unless receivers believe the sender has sent an untrue message. In this case, a receiver broadcasts an untrue attack warning into the VANET towards an RSU so that it can decide on the disagreement.

Loss of trust results from performing several malicious activities in the network. Whenever drivers deceive others, their trust score will be lowered upon the proof of the deceit. With sender-side trust screening, once blacklisted, malicious drivers are prevented from sending messages by the TPD. At this point, they are barred from accessing the services provided by the VANET. This contrasts with existing schemes where vehicles with low trust score can still generate messages, although these will typically be ignored at the receiver once the poor trust level of the sender is discovered. However, this takes time, so vehicles may face traffic jams or congestions unnecessarily. Therefore, VANETs employing sender-side screening will have a lower response time and communication overhead than receiver-side approaches.

1.3 Summary of the Novel Contributions

This thesis has contributed the following novel features with the proposed Tamper-Proof Device (TPD)-based sender-side trust management framework for VANETs:

1. This framework employs sender-side trust management to achieve access control using information accuracy, delay and position difference of the event and the message sending location collected from the sender vehicle itself. Unlike other approaches (Lu et al [6], Wei et al [8], Dahiya et al [10], Haddadou et al [13], and Mühlbauer et al [14]), there is no flow of trust metrics or feedback data unless a reporter vehicle refutes an announcement. This improves communication efficiency. Furthermore, various classes of message, along with their associated trust threshold, are defined for regulating access control. This is used to enable or disable the set of events on the driver dashboard based on his / her trust score. Receivers can immediately act on a message from a sender because the event announcement is regulated by the TPD of the sender vehicle, or they can report an untrue attack if they believe it is untrue, such as based on their own observations. The framework includes a betray field in messages as an additional indication of trustworthiness to the receivers concerning whether the sender has always been trustworthy within a given timeframe. When receivers find a null value in this field, the message can be believed more than a message with a value in this field as it has come from a sender which has not sent any false messages during a specific timeframe, for example, a year.

2. The framework uses a driver profile database to store the results of most recent dispute decisions of drivers. This database does not hold the events originated from drivers but the decisions from previous disputes about drivers as previous reward or punishment decisions are used in the assessment of the reward or punishment in the current dispute.
3. The scheme employs a collaborative untrue message discovery algorithm for detecting various forms of attack. Using this scheme, RSUs rule the validity of disputed events using feedback from the trusted clarifiers and official vehicles. When there is feedback from an official vehicle the collaboration process is bypassed, and a decision is reached immediately. Also, an RSU informs nearby RSUs so that the concurrent evaluations of the same dispute are prevented at different RSUs.
4. A Markov-Chain driver behaviour model is developed for examining the honest/dishonest announcement / untrue attack reporting from various trust states, where each trust state is associated with a range of trust scores. Drivers change trust states when their trust score moves out of the range for the current trust state. Drivers' announcement / untrue attack reporting is governed by a probabilistic distribution from each trust state. The analysis confirms drivers behave more positively when their trust scores are higher, and they belong to a higher trust state and vice versa.
5. To the best of our knowledge, the application of fuzzy logic to an RSU reward / punishment mechanism is unique. We believe no similar application of fuzzy logic has been published. A fuzzy logic-based RSU controller is developed to intuitively determine a fairer level of reward / punishment for the disputing drivers. This scheme replaces the fixed RSU judgement policy and considers driver specific (driver history of rewards / punishments) and event related information (severity of incident and RSU confidence score in the sender / reporter) for reward / punishment assessment.
6. The framework is simulated in the Veins (Sommer et al [17], see https://veins.car2x.org/) to verify its satisfactory operation. The communication overhead and response time are compared against a typical reputation approach (Mühlbauer et al [14]) with varying vehicular densities and speeds. Moreover, the accuracy of the framework is measured in relation to false positive, false negative, true positive and true negative observational data with differing percentages of malicious and benevolent feedback. The fuzzy logic-based RSU application is analysed in MATLAB and then the defuzzified output data are collected considering all possible input values from the fuzzy inputs. These data are then loaded onto OMNeT++ (Varga et al [18]) which an RSU then uses for fuzzy reward / punishment determination.
7. List of Publications:

i. Shahariar, R. and Phillips, C., 2023. A TRUST MANAGEMENT FRAMEWORK FOR VEHICULAR AD HOC NETWORKS. *International Journal of Security, Privacy and Trust Management (IJSPTM), 12*(1), pp 15-36.

ii. Shahariar, R. and Phillips, C., 2023. A FUZZY REWARD AND PUNISHMENT SCHEME FOR VEHICULAR AD HOC NETWORKs. *International Journal of Advanced Computer Science and Applications (IJACSA), 14(6),* pp 1-17.

1.4 Structure of the Thesis

This thesis comprises seven main chapters. Every chapter is briefly introduced with a clear description of the scope of the chapter, and it is ended with a summary of the key points covered in that chapter.

Chapter 2 starts by reviewing VANET characteristics, architecture, components, and communication, Next it considers the security requirements for VANETs including the different types of attacks it may encounter. This is followed by a review of the state-of-the-art security solutions to thwart external attacks in VANETs. After this, we devote a section to first defining trust, and then classifying trust management models along with an exhaustive survey showing how existing trust management frameworks manage trust for security. Next, I identify trends and issues with existing trust management approaches. Finally, we summarise the key elements of this chapter.

Chapter 3 presents the proposed trust management framework for VANETs. It first lists the assumptions for the proposed framework and then the mechanisms for registration, blacklisting, and redemption are considered. The key components used in the framework are then introduced. This is followed by a description of how trust is evaluated inside the TPD considering different networking activities. In this section, the message types a driver and/or vehicle can announce based on trust score are specified. A functional overview of the proposed trust framework is provided. Furthermore, the RSU algorithm is presented for how traffic events are managed. The mechanism for untrue attack detection is presented next together with its associated algorithm. Flowcharts for each component are then given. Finally, the chapter is summarised.

Chapter 4 surveys available network simulators, traffic simulators, and combined simulators in the literature. From these simulators, the Veins simulator is chosen to implement and analyse the proposed trust management framework. Initial setup, communication scenarios and different traffic cases are covered sequentially. This is followed by the implementation details of the proposed trust management framework. Then verification is conducted to confirm the framework can manage the trust as per the design. After this, two tests are performed to confirm the framework can thwart untrue and inconsistent attacks in VANETs. In Chapter 1 I mentioned that the travel time of vehicles will increase if they detour due to an untrue message. However, traffic comfort is improved with accurate messages (i.e. avoiding traffic jams). This is illustrated as the effect of untrue and true messages on the average travel time. The

accuracy of the proposed framework is considered next, and the results are depicted in charts. Finally, the proposed framework is compared with a baseline reputation approach in terms of communication overhead and response time which is also shown on via charts.

Chapter 5 describes a fuzzy logic based RSU reward or punishment controller where the level of reward or punishment is based on various factors. This also includes a detailed fuzzy process description for the proposed extension. Furthermore, a Markov-chain based driver model is also presented. The goal of this model is to better represent driver behaviour. It is used in concert with the fuzzy controller so that the trust score of drivers is encouraged to remain within the normal range as much as possible.

Chapter 6 describes the implementation of the extension of the proposed framework. A comparative analysis with the basic trust management given in Chapters 3 and 4 is presented and discussed. Finally, the thesis is concluded in Chapter 7 with suggestions for the future work.

1.5 Summary

In this chapter the motivation and objectives of this research are clearly stated. In addition, a set of requirements is defined for the proposed trust management framework. After this, the novel contributions of this research are listed. Finally, the structure of this thesis is presented. In the next chapter, VANET security issues and solutions are covered including an exhaustive literature survey of existing trust management frameworks along with their features and communication requirements.

Chapter 2: VANETs, Security Issues, and Trust Management

2.1 Introduction

This chapter first defines what is meant by a Vehicle Ad-hoc Network (VANET) and then describes the security requirements, types of attack, and defence mechanisms for thwarting attacks in a VANET. In literature, some approaches are solely cryptography-based, and some are trust-based. Additionally, some approaches combine trust management with cryptography to improve the overall security of a VANET. As both external and internal attacks are possible, a VANET requires security mechanisms for thwarting both types of attack. A great number of works have been proposed to thwart external attacks. Less attention has been paid to mechanisms for resisting internal attacks; this remains ongoing research. In this chapter, trust management is considered a pivotal mechanism to thwart internal attacks for VANETs. To this end, we exhaustively survey the existing research. It is discovered that existing trust management frameworks suffer from high communication overhead and decision latency. As vehicles collect / generate trust metrics and recommendation data to verify a source and / or event this increases the volume of messages (communication overhead) as well as the time taken to reach a decision. Thus, we consider communication efficiency an important issue to consider when developing a trust model for VANETs. Finally, a new research direction is identified for trust management in VANETs.

2.2 VANET Characteristics

A VANET is a mobile, wireless ad hoc network consisting of vehicles equipped with an onboard unit (OBU) and Roadside Units (RSUs), and optional central authority. The elements typically communicate with each other using the IEEE 802.11p-based DSRC protocol. A dedicated bandwidth of 5.9 GHz is reserved from the WAVE protocol stack for the successful communication of VANETs (Jiang et al [1]). Vehicles and RSUs can broadcast events as needed, but they broadcast beacons periodically. Vehicles exchange beacons for status updates including speed, position, acceleration, and direction (Lu et al [6]). A Central authority (CA) stores records about events, registered and blacklisted vehicles. Also, some research uses the "cloud" with VANETs to store data and to perform calculations.

A VANET is an intermittent and opportunistic network as vehicles meet each other spontaneously. The neighbour list of a vehicle changes very frequently as the topology and distribution change rapidly in VANETs (Ahmad et al [11]). A VANET is different in nature from other wireless networks. In VANET, vehicles move fast (1km/s to 200Km/s), exchange messages within 0~1000m distance, and the topology changes more dynamically than regular ad hoc networks. In this network, one vehicle broadcasts a message, which is relayed by intermediate vehicles to reach other vehicles. That means,

both single and multiple hop communication are common. When there is no vehicle in direct range, then messages are simply dropped. This network requires broadcasting of events at the right time, otherwise, traffic congestion, or other undesirable phenomena may appear on the road.

2.3 Architecture and Components of a VANET

All major types of vehicles including regular vehicles, official vehicles (police, ambulance, fire service vehicles) and public authority vehicles (buses, and licensed taxis) can be part of a VANET. They expect to receive timely traffic updates from the nearby RSUs. Fig. 2-1 depicts a typical VANET where all the key elements are identified.

2.3.1 Regular Vehicles

Regular vehicles are the primary users of a VANET. They broadcast periodic beacons and traffic events. A vehicle announces a traffic event only when it observes one on the road. Vehicles that receive the event are called receivers. Upon reception, they may act on the event or retransmit it to neighbours so that nearby vehicles are informed about the traffic incident. In this way, one vehicle helps neighbouring vehicles from getting stuck in congestion or an undesirable situation on the road. An intermediate vehicle that forwards a message is called a message relayer. Every vehicle is pre-equipped with a transceiver to communicate with other OBUs and RSUs (Hasrouny et al [19]). The OBU may have an Event Data Recorder (EDR), and Global Positioning System (GPS) sensor (Hasrouny et al [19]). It is also common to equip a Tamper-Proof Device (TPD) to hold data records or to perform some manipulation inside the vehicles (Rostamzadeh et al [20], Pournaghi et al [21], and Jing et al [22]).

2.3.2 Official Vehicles

Many types of official vehicles may be present on the road. Among them, police, ambulance, and fire service vehicles are the most frequent. They visit an event location to intervene when they are instructed. When the event is resolved, they typically broadcast a message announcing the resolution so regular vehicles can use that road again. Event information from them can be considered completely authentic.

2.3.3 Road-Side Units (RSUs)

They are placed alongside the road to broadcast timely traffic updates to vehicles. RSUs can communicate themselves either using a dedicated broadband network or using a wireless network. Also, they are connected to the Central/Trust Authority (CA/TA) through a dedicated wired or wireless Internet connection. RSUs send information about traffic incidents to the CA/TA. Vehicles receive periodic traffic updates as well as emergency events from RSUs (Lu et al [6] and Hasrouny et al [19]). Additionally, RSUs treat messages from official vehicles as “high priority” when they are attending specific emergency events.

2.3.4 Trust Authority (TA)

The Trust/Central Authority (TA/CA) is the ultimate authority in a VANET. The TA registers vehicles/RSUs, authenticates vehicles, and blacklists malicious vehicles when the extent of their malicious activity exceeds a threshold of poor behaviour. It is mandatory to place the TA in a highly secure environment. Furthermore, the TA must be equipped with sufficient computing resources to fulfil the demand of processing requests from other entities.

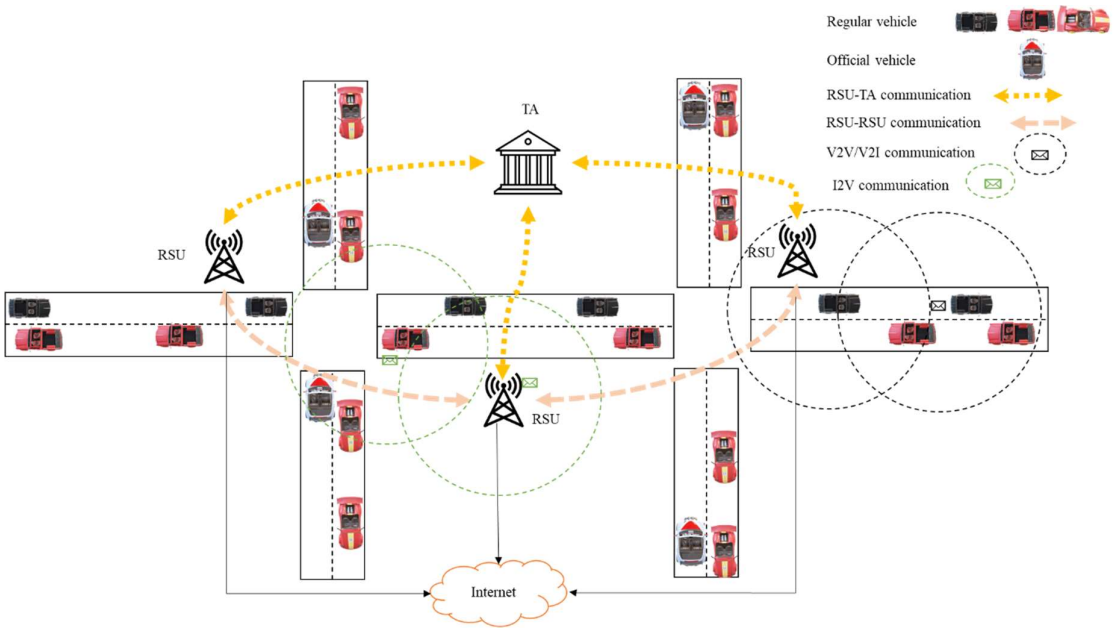


Fig. 2-1: A VANET with its Key Components

2.4 VANET Communication

In VANETs, when a vehicle notices an event, it broadcasts a message and other vehicles relay the message onwards until specific conditions are met. When this event reaches an RSU, it also periodically broadcasts the message until the event is resolved. In some severe cases, RSUs share traffic events with neighbour RSUs so that more vehicles will not enter the problematic area from the neighbouring regions. In this way, severe traffic chaos can be avoided. Vehicles and RSUs also send beacons using single-hop communication. In contrast, traffic events are announced using multi-hop communication. Multihop communication facilitates communication among vehicles outside of the transmission range of the originator. In this situation, intermediate vehicles simply forward the messages (Studer et al [23], Huang et al [24], and (Aquino-Santos et al [25]). Additionally, group communication is common inside a cluster in a VANET.

2.5 Security Requirements for VANETs

The security requirements for VANETs are: availability, authentication, access control, privacy and confidentiality, integrity, real-time constraint or efficiency, and non-repudiation (Engoulou et al [26]).

An attack is a threat to a system that enables an attacker to access, modify, add, delete, or reveal information without consent from the authority (Arif et al [27]). Attacks hamper the normal operation of VANETs. An attack can be of two types which are active and passive. An active attacker modifies the data of a target entity or changes the ongoing route to the target (Upadhyaya et al [28]). In contrast, the passive attacker only monitors the network activities to reveal information to find vulnerabilities (Upadhyaya et al [28]). Also, attackers can launch attacks from both inside and outside. An attacker is an outsider when he is not authorized in the VANET. An outsider can eavesdrop on user communication in the network to launch Distributed Denial of Services (DDoS), black hole, or false information injection attacks in a VANET (Zhou et al [29]). An insider attacker is an authorized user of a network who can launch all possible forms of attack as he knows about the network. Insider attacks cannot be thwarted using only cryptographic schemes which work against outsider attacks (Kerrache et al [30]). Hence, trust management is applied to limit internal attacks. Trust management can classify honest and dishonest vehicles as well as revoke messages from malicious vehicles (Dahiya et al [10] and Ahmad et al [11]). Furthermore, a trust scheme can achieve access control, mischievous vehicle isolation and allocate punishment for mischievous actions (Gazdar et al [12]). Despite this, trust schemes cannot protect VANETs completely (Tangade et al [7]). It is not assured that a trusted vehicle will always announce trustworthy messages. Hence, an evaluation method using predefined metrics can determine the reliability of relayed information, and reward or punish a sender vehicle appropriately (Gazdar et al [12]). By evaluating announcements, trust management can play a major role in identifying malicious vehicles and protecting the VANET from mischievous actions. Although, initially, trust management is applied in social science (Butler [31] and Mayer et al [32]). It is now used in networking applications (Kerrache et al [30]). Before highlighting trust management in detail, it is useful to consider the types of attack that can be launched in a VANET.

2.5.1 Threats to Availability

The term availability signifies a system is in a functioning state though a part of the system may be malfunctioning or faulty (Kumar et al [33]). Channel availability is a mandatory requirement for communication in VANETs, otherwise, messages cannot be transferred between entities. Thus, availability must be satisfied for every element in VANETs. Below, is a list of attacks which may affect the availability:

- Denial of Service (DoS) attack (Lu et al [6]): DoS is the most well-known attack a VANET can suffer from which prevents legitimate users from getting access to resources to which they are entitled. Both insider and outsider perpetrators can jam the wireless medium and block any networking resources granted to legitimate users. Another variant of this attack is called Distributed Denial of Service (DDoS) attack when the attackers launch an attack in a distributed manner to disrupt normal network operations.

- Jamming attack (Lu et al [6]): The attacker uses the same frequency and a stronger signal than the data signal during normal communication to make the medium busy.
- Black hole and grey hole attack (Lu et al [6]): Blackhole attackers deny forwarding of packets or forwarding in another direction to disrupt the established connection. Initially, grey hole attackers behave honestly to gain the trust but later they drop packets to take advantage.
- Malware attack (Lu et al [6]): Whenever a vehicle or an RSU is infected with malware and behaves abnormally this is called malware attack.
- Broadcast tampering attack (Lu et al [6]): Authorized users insert incorrect data with the original data to hide the actual meaning of the warning message from other authorized users.
- Greedy behaviour attack (Lu et al [6]): In this attack, attackers abuse the shared bandwidth.
- Spamming attack (Lu et al [6]): An attacker broadcasts spam messages in the VANET which collide with genuine messages to confuse recipients.

2.5.2 Threats to Authentication

Authentication is the verification of a user when attempting to get access to the system resources to which they are entitled. Authentication is the first level of defence against any kind of intrusion. For VANETs, three things are typically authenticated which are: ID authentication (license plate, chassis number), property authentication (sender is a car or RSU), and location authentication (Kumar et al [33]). Attacks that interfere with authentication are:

- Sybil attack (Lu et al [6]): In this attack, perpetrators create multiple fabricated identities and then spread false messages using different identities.
- Node impersonation attack (Engoulou et al [26]): An attacker can send messages using the identity of another vehicle. When messages are received from a real user, he/she modifies and resends them with a fake or stolen identity for his/her own benefit.
- Tunnelling attack (Lu et al [6]): In this attack, an attacker creates a long-distance tunnel to communicate with a targeted remote vehicle and treats it as its neighbour.
- GPS spoofing attack (Lu et al [6]): In this attack, an attacker generates a false GPS signal to deceive other entities.
- Key and/or certificate replication (Hasrouny et al [19]): The attacker utilizes the key and/or certificate of another vehicle to authorize itself.

2.5.3 Threats to Privacy

Privacy is one of the greatest challenges for a VANET to be successful (Engoulou et al [26]). Drivers want to keep their private information secret (Karnadi et al [34] and Samara et al [35]). Privacy protection means the protection of location and identity information against profiling. Confidentiality ensures only the intended receiver can have access to the desired information. Thus, it is necessary to allow anonymous communication besides trusted announcements in VANETs (Dhurandher et al [36]).

The objective is to develop a system where sensitive information cannot be accessed unauthorizedly (Engoulou et al [26]). Attacks against privacy and confidentiality are summarised as:

- Eavesdropping (Lu et al [6]): is the listening to any confidential information by an unintended entity.
- Man-in-the-middle attack (Kumar et al [33]): An attacker listens the communication between a sender and a receiver and then inserts modified messages. Attackers confuse both the sender and the receiver to believe that they are communicating directly, unaware of the modified messages.
- Home attack (Kumar et al [33]): There are three ways an attacker can launch a home attack. First, an attacker takes control of an OBU of a vehicle and then injects wrong messages in the VANET. Second, an attacker takes control of the sensor of a vehicle to modify the behaviour of the sensor. Third, an attacker takes control of the Electronic Control Unit (ECU) of a vehicle and changes the speed of the vehicle.
- Social attack (Kumar et al [33]): An attacker can persuade a driver to broadcast inappropriate messages so that other drivers get annoyed, and their driving behaviour is affected.
- Identity disclosure attack (Arif et al [27]): An attacker sends a message to a target vehicle legitimately to obtain a reply from the target vehicle and to track it. Then the attacker discloses the identity and the location of the target vehicle.
- False information injection attack (Lu et al [6]): An attacker spreads false information so that other vehicles make inappropriate decisions.
- Traffic analysis attack (Lu et al [6]): An attacker listens and analyses transmitted messages from others to capture confidential information.

2.5.4 Threats to Integrity

Integrity confirms that the intended data in transit is not altered by any means and the receiver receives whatever the sender sends. Integrity protects data from unauthorized creation, dismantling, and alteration (Engoulou et al [26]). Integrity ensures an attacker cannot alter the meaning of a message so that the message is reliable (Papadimitratos et al [37]). To ensure integrity, a digital signature is first created then it is attached to the message by the sender. Once the message is received by receivers, they verify the digital signature to prior to accepting the message. Integrity attacks comprise:

- Message suppression / fabrication attack (Lu et al [6]): In this attack, intermediate nodes modify messages towards the destination to mislead the intended recipient. As a result, some vehicles may detour using a longer route or may get stuck in a traffic jam.
- Message falsification attack (Engoulou et al [26]): This attack occurs when inaccurate information is disseminated. For example, an attacker sends “an incident on road” message when there is none to mislead vehicles.

- Masquerading attack (Lu et al [6]): In this attack, an attacker uses misappropriated information to get access to the VANET and then sends a false message.
- Replay attack (Lu et al [6]): This attack occurs when an attacker reinjects earlier messages to confuse others.
- Hardware tampering (Engoulou et al [26]): This attack results from interfering with the hardware of an entity in the VANET.

2.5.5 Nonrepudiation

Nonrepudiation confirms that either party involved in the communication cannot deny their participation (Engoulou et al [26]). For example, sender nonrepudiation ensures that a sender cannot later deny the fact that it sent a message.

- Repudiation attack (Lu et al [6]): Whenever a sender or a receiver denies sending or receiving a message it is called a repudiation attack.

2.5.6 Real-Time Constraint / Efficiency

In VANETs, vehicles usually move on planned route to reach their destination. However, when there is an event, for example, congestion or accident along their route, they need to detour, otherwise, they experience a traffic jam / congestion. If this information isn't disseminated and acted upon in a timely manner the severity of the situation may intensify as more vehicles will queue up around an event. Thus, any VANET application should provide prompt decision–making. As verification of an event increases latency, drivers from nearby regions may enter the event zone which results in further congestion.

Event announcements should reach other entities in real-time, so decisions can be made promptly. Sometimes, these messages lead to further network analysis to reach a decision. Thus, prompt dissemination and relaying of event announcements to others is crucial. While managing an event, how many messages are generated and how much time are elapsed to decide on an event, are two important performance criteria. For example, if drivers need to wait 30-45 seconds to decide, then the driver may cross the junction / entrance of the alternate path through which the possible detouring could be made. In this case, vehicles may queue up around an event from multiple directions even if detour options are available. Additionally, vehicle speed needs to be considered when designing a security application for VANETs.

In VANETs, untrue events can be announced besides trustworthy events. So, it is necessary to differentiate between trustworthy and malicious messages. To this end, security and trust approaches employed in VANETs aim to ensure trustworthy message dissemination. However, when an authorised user sends untrue information, security approaches cannot detect these. This false information needs to be identified and the sender punished. A trust model evaluates these events and manages the trust of the sender and / or events. A trust evaluation scheme also spreads approach-oriented messages into the VANET which increases the communication overhead. When they trigger further communication after

messages arrival, this adds latency while deciding on an event. Most existing trust approaches require a considerable decision time and exhibit high communication overhead as they evaluate messages at the receiver side after message arrival. A trust scheme which can manage trust of events and / or a sender with low verification time and communication overhead is more desirable. The main motivation of this research is to develop a new trust model which reduces communication overhead and driver decision time. If trust can be managed at the sending-side, then receivers are relieved from trust verification at run time. This offers nearly zero decision delay (ignoring processing at the receiving side) to the drivers which is advantageous over existing trust approaches. Consequently, drivers can decide quickly on whether they need to detour or not to avoid the probable congestion. Additionally, a VANET should employ a system which prohibits the dissemination of untrue information as early as possible, or it should send the corrective message as soon as it finds out the validity of an event.

2.6 Security Approaches

Security mechanisms protect the availability, authentication, privacy, trust, and non-repudiation through a cryptographic algorithm. Many security approaches have been proposed for VANETs. In this section, some of these mechanisms are discussed briefly.

2.6.1 Cryptography

Cryptography and digital signatures are used to block external attacks. Most of these mechanisms protect authentication, privacy, confidentiality, and integrity. Cryptographic schemes achieve message hiding through encryption and decryption (Karimireddy et al [38]). An encryption algorithm converts the message (plaintext) into ciphertext, and a decryption algorithm converts the ciphertext back into plaintext. Public key cryptography, symmetric key cryptography, and hashing are the most common among the cryptographic solutions.

2.6.1.1 Asymmetric (Public) Key Cryptography

Security approaches based on public key cryptography use public and private keys. The Public Key Infrastructure (PKI) stores the public keys, and the owner obtains a secret key (De Fuentes et al [39]). The certificate authority of the PKI system manages the pair of keys for each registered entity and does the mapping of public keys to entities using certificates (De Fuentes et al [39]). Keys are generated using a cryptographic algorithm, for example, Diffie Hellman, Rivest–Shamir–Adleman (RSA), or Elliptical Curve Cryptography (ECC) (Chandra et al [40]). Vehicles use them to communicate with other vehicles and RSUs. DSS requires excessive computation and memory resources in key exchange and digital signature generation (Karimireddy et al [38]). Vehicles can register online with the CA via RSUs or register offline directly with the CA (De Fuentes et al [39]). In public key cryptography, whenever a sender “Alice” wants to send a message to the receiver “Bob”, then Alice uses the public key of Bob for encryption and only Bob knows his private key. Then, Bob can decrypt the received

message from Alice to get the intended meaning. Mainly, public key cryptography is used for encryption and authentication. The privacy-preserving authentication scheme in Li et al [41] uses public key cryptography to generate pseudonyms to address non-repudiation by tracing the real identity of a vehicle from a third party.

2.6.1.2 Symmetric (Private) Key Cryptography

In symmetric key cryptography, one shared key is used for encryption and decryption. Advanced Encryption Standard (AES), Data Encryption Standard (DES), triple DES (3DES), and Blowfish are well-known symmetric key cryptography algorithms (Chandra et al [40]). Symmetric key cryptography is faster than asymmetric key cryptography which is why it is used for bulk encryption. The key size is smaller although it provides less security as compared to asymmetric key cryptography (Karimireddy et al [38]). The main limitation of the symmetric key algorithm is to maintain a private communication link between the parties to have access to the shared secret key. Reference Sakhreliya et al [42] combines PKI-based symmetric key cryptography with Message Authentication Code (MAC) for VANETs to reduce the authentication delay compared with existing approaches. Also, the vehicle registration number is used as a key so that the owner cannot change it.

2.6.1.3 Identity-Based Cryptography

In Identity-Based Cryptography (IBC), an email address, location, or a telephone number are used as the user's public key. In this authentication system, PKI certificates are not used to reduce the communication overhead and Certificate Revocation List (CRL). In Shim et al [43], the researchers preserve conditional privacy by mapping each message to a unique pseudo-identity. The trusted authority can recover the real identity of a vehicle from a pseudo-identity and RSU can verify multiple signatures concurrently. This approach with the identity-based signature reduces the signature verification time compared to a baseline. In Bradai et al [44], the authors secure the privacy of location and identity data through encryption as well as the non-repudiation of vehicles using IBC. In this approach, the TA assigns a Pseudo-Identity (PID) to each vehicle which is updated periodically. Vehicles use their PID for authentication with RSUs and other vehicles. With this scheme, an adversary cannot collect any meaningful information about a compromised vehicle as it only stores public information in the storage site. It also achieves nonrepudiation by tracing the cause of the incident from the secret key and transferred messages between RSUs and vehicles. In Bhavesh et al [45], vehicles are given multiple levels of anonymity based on their requirement. The scheme controls the level of anonymity of a vehicle by issuing varying number of pseudonyms and attaching different lifetime to each pseudonym. The protocol uses identity-based signature and pseudonyms to achieve anonymous authentication, sender nonrepudiation, and integrity.

2.6.1.4 Hybrid Cryptography

Sometimes asymmetric and symmetric key cryptography are used in combination to improve the security efficiency in networking applications. This hybrid system is designed to maximize the advantage of each approach in one cryptography algorithm. For example, Karimireddy et al [38] combines both RSA and AES to secure communication in a VANET. In Wagan et al [46], the authors propose a low-latency security framework using hybrid cryptography to achieve fast and secure communication.

2.6.1.5 Certificateless Signature Scheme

In Al-Riyami et al [47], a certificateless approach is proposed. In a certificateless signature scheme, the cost of PKI certificates is reduced by using a different third-party Key Generation Centre (KGC) (Sheikh et al [48]). This KGC is responsible for assigning private partial keys. The secret value a user generates consists of actual and partial private keys which in turn generates the public key. In Horng et al [49], the researchers also use a certificateless scheme to convert a traffic message into a pseudo-identity to preserve conditional privacy, but the real identity is traceable by the authority. Cui et al [50] proposes a certificateless aggregated signature using Elliptic Curve Cryptosystem (ECC) to secure conditional privacy between vehicles and the infrastructure.

2.6.1.6 Group Signature

Vehicles sometimes form a group to ease communication among themselves. Within a group, they share a common key to generate messages. Only a genuine entity can reveal secret information. The group can verify the signature of a potential member who sends a signed message. A group signature can promote privacy for VANETs. In Karimireddy et al [38], a group signature is used for distributed key management and to secure the VANET by revoking malicious vehicles, thereby protecting location privacy. Local RSUs share group private keys. Also, Qu et al [51] preserves vehicle privacy using a group signature scheme. The group signature-based authentication approach in Sun et al [52] updates the secret key of a vehicle by the sub-region head and only the regional head can view the key. However, group signature suffers from the meet-in-the-middle attack, forgery attack, and conspiracy attack (Agarwal et al [53]). A meet-in-the-middle attack is used for signature forgery on a combined digital signature method which reduces time than the exhaustive attack. An exhaustive attack is a kind of brute-force attack, for example, considering all combination of passwords. A forgery attack enables a user to send an unwanted request to a system where they are already authenticated when it is already authenticated to reduce its trust. In conspiracy attack, group members try to impersonate other signature by obtaining the secret private key. In Malina et al [54], group signatures are combined with short-term link-ability and batch verification.

2.6.1.7 Cryptographic Hash Function

A cryptographic hash function is used to verify a message's integrity with no encryption. A hash value is attached to the message from the sending entity. The simplest form of hashing is to use a pseudo-number to generate a fixed-size hash value or digest for a given message (Karimireddy et al [38]). Hashing maintains a hash table where the digest works as the index. Hash Message Authentication Code (HMAC) is a type of Message Authentication Code (MAC). HMAC is widely used for both message authentication and data integrity using a cryptographic hash function (MD5 or SHA-1) and a shared secret key (Hu et al [55]).

2.6.2 Confidentiality and Privacy Preservation

Confidentiality confirms that sensitive information remains private or secret. In Sun et al [56], shared key encryption is used to protect the confidentiality of private information and ensure tracking of a vehicle is done legitimately. Lin et al [57] preserves the integrity of data where RSUs are only deployed in the busiest areas. Receivers recompute the MAC using the session key and compare it with the MAC from the sent message to verify the integrity of data. In Raya et al [58], an anonymous public key is used to protect privacy and key exchanges are done in a way such that receivers cannot track the vehicle owner. In Elmahdi et al [59], the researchers thwart attacks against integrity in VANETs using hamming code with a Compression Sensing Data Compression (CS-DC) scheme. This scheme detects and drops modified and injected data from compromised vehicles before forwarding.

Users need to protect their private data so that attackers cannot have access to the sensitive information. In VANETs, user credentials, vehicle location and route data are sensitive data. In most cases, pseudonyms are used, and they are changed regularly by the authority. In Choi et al [60], the authors combine symmetric key cryptography with short-term pseudonyms to protect vehicle privacy. Whenever a message is received, its MAC is verified by using the symmetric session key.

2.6.3 Authentication

Authentication is the process of verifying an entity whenever it wants to gain access to any network resources. In VANETs, both vehicle and message authentication are required (Agarwal et al [53]). Message authentication verifies the sender of a message whereas, vehicle authentication confirms a particular vehicle is involved in the communication. A basic authentication scheme involves signing a message by the sender and then verifying this at the receiver. Examples of authentication schemes are HMAC, digital signatures, and certificates (Kaur et al [61]). In Jiang et al [62], the researchers propose an IBS-based batch authentication scheme by using a HMAC while verifying the certificate and the signature. This approach uses pseudonyms for preserving privacy. In Li et al [41], IBS is used to authenticate vehicles and RSUs. Also, ID-Based Online / Offline Signature (IBOOS) is used to reduce communication overhead in vehicle-vehicle authentication. In Zhang et al [63], the public key is used

as an ID to avoid certificate management and the private key of a signature is used only once which is associated with an identity. This approach combines a signature and addresses for privacy-preserving authentication. In Chim et al [64], the information source is authenticated when drivers ask for route information from others. Using anonymous credentials, the privacy of drivers is preserved, and the TA can trace the real ID of the vehicle (Chim et al [64]). The researchers in Lu et al [65] propose a Dynamic Key Management Scheme (DIKE) for Location-Based Services (LBS) which not only achieves privacy-preserving authentication but also detects duplicate registration of vehicles. Lin et al [66] uses a token-based authentication scheme for VANETs. In Chuang et al [67], the authors propose trust extended authentication approach by using XOR and hash function.

2.6.4 Channel Availability

Reference Shukla et al [68] proposes a congestion control algorithm to manage the communication channel for VANETs. It invokes congestion control to freeze all the MAC queues except the queue being used for the event announcement. In Cardoteet al [69], the authors worked on a multi-hop connection period to investigate the path stability timescale for highways. They determined the physical connectivity times among the relay nodes using a Gaussian distribution-based speed distribution and exponential distribution-based inter-arrival time. In Kumar et al [70], the link availability probability is determined when comparing the performance of three routing protocols namely DSR (Johnson et al [71]), AODV (Perkins et al [72]), and FSR (Gerla et al [73]) for an urban scenario. Ali et al [74] proposes a decentralized authorization scheme for clusters. The CA is formed from the highly trusted nodes in a cluster. This method confirms the enhanced availability of security services as certificates that can be issued even if the CA is unavailable. In Okamoto et al [75], the researchers use the push and pull method to control the distribution of location-dependent data towards higher density areas. In Park et al [76], the researchers improve the data accessibility of vehicles using a copy of data from RSUs.

2.6.5 Access Control

Access control defines the roles and privileges in a network (Engoulou et al [26]). Access control limits the use of different components and determines a user's accessibility to network resources. Its implementation is achieved through system-wide policies which include defining actions for every component in the network and defining accessibility of users based on their role. Authorization is a part of access control and defines the rights of each vehicle in the network (Qian et al [77]). For example, some messages from an official vehicle can be hidden from normal vehicles. Another example of access control is the exclusion of malicious vehicles from a VANET by blacklisting. In Moustafa et al [78], the researchers implement an access control mechanism for allocating services to highway vehicles. With this scheme, they assign IP addresses to each mobile client. They present a Kerberos model where clients present credentials to get a ticket for every service, and with this ticket, the client obtains service access.

2.6.6 Certificate Revocation

One of the roles of the CA/TA is to assign certificates and keys in the PKI. If the CA/TA detects a vehicle is sending inappropriate messages or spreading incorrect information in a network repeatedly, then the CA/TA can revoke the certificate of the vehicle and can mark it as “blacklisted”.

2.6.7 Blockchain

Blockchain was first used in digital cryptocurrencies called bitcoin (Nakamoto et al [79]). Nowadays, it is also used to secure network communications because of the high reliability it can provide. As blockchain is naturally tamper-proof, it is used to secure data in RSUs (Jiang et al [80]). The main part of a blockchain is the decentralized distributed database which matches the architecture of the VANET (Chen et al [81]). Using blockchain, authentication and trust can be enriched and all RSUs have the same copy of blockchain (Dwivedi et al [82]). All members of a blockchain have equal weights. Blockchain applies a consensus mechanism to add a block of information and each block is timestamped. Blockchain is based on Markle tree which maintains an ordered list of blocks and each block contains a specific number of transactions. Each block is chained with the previous block by preserving a nonce (a counter that can be used only once within the blockchain) and hash of the previous block. In this way, blockchain achieves immutability. There are three main categories of blockchain (Nakamoto et al [79]). The first one is called the permissionless (public) blockchain in which anyone can join the network and can write or add information to a blockchain. There is no central authority in this type of blockchain. Another one is called the permissioned (closed) blockchain in which there exists a central authority that grants access to the blockchain. The last one is the consortium blockchain which is neither permissionless nor permissioned but lies between them. Only the “miner” can distribute blocks to others. There exist several miner selection algorithms to elect a miner which are proof-of-work, proof-of-stake, proof-of-capacity (Yang et al [83]). Blockchain is used in VANETs for authentication, privacy preservation, trust establishment, event management, and smart contract-based system so far (Nakamoto et al [79]). The successful deployment of blockchain depends on the active involvement of vehicles so that other vehicles may have time to decide on the context, for example, whether to detour.

2.6.8 Detection of Malicious Data

In VANETs, vehicles require trustworthy and accurate information to ensure safety. The sender must be a legitimate user. A VANET must employ a scheme to detect malicious information sent over the network, to warn vehicles on time, and to disregard the false information as early as possible. Also, it should punish the wrongdoer to demotivate mischievous action.

Vehicles in VANETs verify the messages and / or sources to determine the reliability of the message. In this way, vehicles detect an untrue message disseminated across the network from a malicious vehicle. This malicious information may cause some vehicles to detour and in some cases, they may be

queued up around an event which results in traffic congestion. Consequently, this affects the traffic comfort of drivers who believe the untrue information. A trust model and security are used to thwart this type of attack. Mostly, approaches run a false message detection scheme after the arrival of messages. Also, several approaches have been proposed to limit the dissemination of false data in VANETs (Ahmed et al [84] and Hu et al [85]). While verifying traffic events, some approaches collect recommendation / feedback at RSUs, while others collect data at receiver vehicles. When an RSU collects recommendations, it disseminates a corrective message after the detection of malicious data. If vehicles collect recommendations, they individually determine the trustworthiness of an event based on the collected information. However, as RSUs have broader knowledge and they are fixed in the event zone, they can collect more messages than vehicles which leads to more accurate detection of false data than vehicular detection schemes.

2.6.9 Hardware Security

Security can be provided at a hardware level. For example, Chim et al [86] uses a Motor Vehicle Event Data Recorder (MVEDR) module to record all messages disseminated from the vehicle. Some approaches also use a Tamper-Proof Device (TPD), or Tamper-Proof Module (TPM). The TPD (or TPM) is a hardware unit, typically installed inside of vehicles to confirm privacy and security. The TPD communicates with the embedded sensors and other hardware units using software. They are supposed to be tamper-resistant so that some information can be stored and updated securely inside (De Fuentes et al [39] and Raya et al [87]). Reference Sumra et al [88] proposes a TPM-based solution to achieve security, trust, and privacy for VANETs. In Rostamzadeh et al [20], Pournaghi et al [21], and Jing et al [22], the TPD is used to generate and receive encrypted messages. Some trust management approaches like the FACT framework (Rostamzadeh et al [20]) also consider locating a TPD in every vehicle. In Tangade et al [7], a TPD is used for storing major system keys at RSUs. Also, conventional security protocols can be implemented at the circuit level. However, an integrated circuit overbuilder can insert extra circuitry while building the security module in a factory which is called a hardware trojan (Kaur et al [61]). Another method called reverse engineering is also a threat to security. One can use it to understand the workings of the security module and then can bypass them.

2.7 Trust

Trust is a key facet of security which is defined as "A system or component that behaves expectedly for a particular purpose" (Pearson et al [89]). Trust is the confidence that one vehicle places in the actions of another vehicle. In VANETs, trust is established between vehicles based on the messages they exchange with each other over time. Receiver vehicles calculate the trust of a sender and its messages based on some predefined metrics, for example, past interaction(s) with the sender, recommendations, and the reputation of the vehicle (Tangade et al [7]). In addition, trust establishment

and trust evaluation are extremely difficult because many factors need to be considered in a short time (Guleng et al [15]).

2.7.1 Trust Management for Vehicular Security

A VANET disseminates critical messages (accidents, traffic jams etc) among vehicles and other entities. Thus, VANETs demand a secure, trusted environment for the dissemination of accurate, reliable, and authentic information. It is extremely difficult to maintain this requirement because of the large scale and open environment which is susceptible to various attacks. A cryptographic approach cannot protect the VANET completely due to attacks from authorized users. To thwart this kind of attack, trust building has been adopted (Tangade et al [7], Wei et al [8], and Tangade et al [9]). Trust and reputation enrich security by encouraging good behaviour and penalizing malicious behaviour (Hussain et al [90]). In many ad hoc networks, it is possible to wait and collect some interaction data before inferring the trust of the node. This is not feasible with a VANET as vehicles have brief and infrequent interaction cycles (Hussain et al [90] and Huang et al [91]). The communications between unfamiliar neighbours are more likely in VANETs (Hussain et al [90]). Hence, trust propagation is no longer an appropriate solution. It is important to evaluate the trust of a vehicle quickly or almost immediately as their interaction is brief (Hussain et al [90]).

2.7.2 Classification of Trust Approaches

Trust approaches aim to classify honest and dishonest vehicles. Some approaches also adopt blocking of malicious vehicles when their behaviour is proven to be extremely bad. There are several ways trust approaches can be classified. For example, some approaches store the trust of vehicles in a centralized server whereas others compute trust at the vehicles in a decentralized manner. Also, they can differ in their data collection mechanism or the technique they use for trust management, for example, blockchain, machine learning, fuzzy logic, or probabilistic and statistical mechanisms.

2.7.2.1 Centralized versus Decentralized Schemes

Trust approaches, such as Li et al [92], Li et al [93], and Li et al [94] follow a centralized architecture as they have a centralized server. The centralized server acts as the trust or reputation server. This server collects information about an event from neighbouring vehicles and then updates the trust of the sender vehicle using a predefined method. These approaches follow a good practice by storing trust / reputation in one place safely. However, this type of approach periodically sends the updated trust to the vehicles which needs excessive trust metrics dissemination and burdens the VANET. In some cases, a receiver vehicle can ask the centralized server about the sender vehicle's trust to verify received messages. This type of communication adds to the driver decision time which is not suitable for VANETs as drivers need to take decisions promptly, otherwise, drivers may cross the junction point missing an alternative

route to avoid the traffic jam. As a result, congestion will be more common on roads if this type of trust model is employed. Alternatively, some approaches evaluate and manage trust in a decentralized manner (Mrabet et al [95], Pu et al [96], and Chen et al [97]). Trust is evaluated by the receiver vehicles, and they exchange trust records and / or historical interactions with neighbours and RSUs. Most of the blockchain-based trust models are decentralized. These approaches offload the computation from the RSU. However, the exchanged messages can stress the network as they exchange direct and indirect trust messages in addition to event announcements.

2.7.2.2 Blockchain-Based Schemes

Blockchain-based trust models adequately improve the security and trust in VANETs. Many trust approaches use blockchain as a key repository of tamper-proof information for trust management (Yang et al [83], Li et al [94], Mrabet et al [95], Pu et al [96], Chen et al [97], Li et al [98], Haddaji et al [99], Luo et al [100], Yang et al [101], Xie et al [102], Yang et al [103], and Huang et al [104]). The main advantage of blockchain is the ability to maintain consistency of information by sharing the same copy of the blockchain among the RSUs. Every block in the blockchain is managed and updated by the RSUs. However, during block insertion, a fake block can be inserted by a fake RSU. Consensus mechanisms, assuring block verification and validation, need more research as suggested by Dwivedi et al [82]. Day by day vehicles are increasingly seen on road which influence to use multiple blockchains to fit the increased demand. Dwivedi et al [82] suggests designing a suitable framework to address the scalability issue. It is also stated in Dwivedi et al [82] that some models run miner selection at RSUs whereas, in other models, the vehicles run the miner selection algorithm. There should be a standard to state which entity should do it in the blockchain.

2.7.2.3 Artificial Intelligence-Based Schemes

Some approaches employ Artificial Intelligence (AI)-based trust computation, for example, Huang et al [104], Malhi et al [105], Sharma et al [106], and Mankodiya et al [107]. AI is used to perform the trust computation using, for example, deep learning (Tangade et al [108]), reinforcement learning (Guleng et al [15], Zhang et al [109], and Guo et al [110]), decision trees (Sharma et al [106] and Mankodiya et al [107]). Fuzzy logic (Guleng et al [15] and Malhi et al [105]) is also used to handle uncertainty in VANETs. However, the application of these techniques has some limitations. For example, in Tangade et al [108], deep learning is used by the RSU to verify a sender and its messages, and sender's trust is computed by the trusted entity. When receivers receive this event, they need to wait for the verification from the RSU. So, driver decision time increases which causes them to be queued around an event. This may create driver frustration. Also, fuzzy logic (Guleng et al [15] and Malhi et al [105]) based trust mechanisms require repeated sensing of messages from neighbours. Furthermore, decision tree-based approaches (Sharma et al [106] and Mankodiya et al [107]) only detect fake position attacks from basic safety messages. The application of reinforcement learning is limited

to finding a factor or adjusting a factor in relation to the trust computation. Therefore, the application of machine learning in trust management for VANETs has so far failed to achieve the VANET strict timing requirement and / or suffers from communication overhead. However, we believe careful implementation of any of these algorithms can achieve beneficial results.

2.7.2.4 Data Collection

Trust information gathering methods are classified using the distance from the source to the data collector. They can be classified into three main categories which are direct (Li et al [98], Shaikh et al [111], and Wu et al [112]), indirect, and hybrid (Guleng et al [15], Wei et al [113], Najafi et al [114], Mármol et al [115], Zhou et al [116], and Dotzer et al [117]) based on the nature of the data collection (Wei et al [113]). Methods that rely on direct trust data, collect information only from the one-hop neighbours. Indirect trust data is collected from the recommendation of one-hop neighbours about the non-neighbouring nodes. Indirect trust-based model is rare in the literature, but this concept often forms part of hybrid data collection schemes. These models evaluate trust by gathering data which increases communication overhead and driver decision time in some instances. However, this is a well-established classification technique followed by most researchers for VANETs.

2.8 State-of-the-Art Trust Models

Till now most trust models have been proposed for mobile ad hoc networks (Verma et al [118]), and wireless sensor networks (Abdelwahab et al [119]). However, they cannot be directly applied to VANETs due to their nature and requirements. Trust model for VANETs remains an active research topic though many trust models have been proposed (Tangade et al [7]). These models can be categorized into three groups based on their evaluation nature. The first group is called Entity-Oriented Trust Models (EOTM) which only verify an entity's trust. The second group is called the Data-Oriented Trust Models (DOTM) which evaluate the credibility of data. Hybrid Trust Models (HTM) evaluate both an entity's trust and the reliability of the data. In the next subsections, trust models are briefly reviewed with their strengths and weaknesses.

2.8.1 Entity-Oriented Trust Models

In this section, some existing entity-oriented trust models are reviewed. Entity-oriented trust models are typified by Haddadou et al [13], where the researchers securely manage allocated credit using a TPM on every vehicle. A vehicle first gets the transmission cost and the signed message from its TPM. Receiver vehicles consider the sender's reputation to trust the message and the trust is revised using feedback from all receivers. This approach considers the presence of false attacks and benevolent vehicles. However, the process of setting a revised trust score can lead to excessive communication. In Siddiqui et al [120], the researchers consider familiarity, packet delivery ratio, timeliness, and interaction frequency to manipulate a weight-based aggregated final trust. They also analyse the time-

aware trust of vehicles based on varying recent histories of interactions. However, they do not consider any attacker model for validation.

In Uma et al [121], trust is computed from past experience, neighbouring vehicle information, trust of the vehicle, and the packet delivery ratio. This approach has a trust manager, route manager, and decision manager. The trust manager finds the path trust and the time required to forward a message to the destination. The decision manager decides whether the vehicle would participate in the packet forwarding function. If this condition is failed, the decision manager updates a nearby RSU about the maliciousness of the vehicle. This model selects a path with the highest trust and lowest delay. This approach is validated in ns-2 (https://www.isi.edu/nsnam/ns/index.html [122]) and analysed the packet delivery ratio, delay, and the number of routes. However, they only achieve trusted routing. Reference Zhang et al [123] builds a stable trust link graph from the local trust of vehicles and then finds the global trust using a TrustRank algorithm. First, local trusts are calculated using Bayesian inference from neighbour recommendations. This local trust data is forwarded to the TA via RSUs to produce the trust link graph. After this, the TA finds the global trust and sends it back to vehicles via RSUs. They also consider driver, vehicle, and behaviour factors in finding a "seed" vehicle and untrustworthy vehicles. The trust value of vehicles is then transferred from the seed vehicle to the normal vehicles in a Markov process manner using the trust link graph. This approach is compared with one existing approach to explore its effectiveness in quickly isolating malicious vehicles. Also, robustness against newcomer, on-off and collusion attacks is better than the baseline approach. However, this model suffers from the network-wide local trust data collection and global trust data dissemination.

In Saraswat et al [124], an Analytical Hierarchy Process (AHP) based trust computation scheme is proposed for VANETs. In this model, the Perron-Frobenius theorem-based direct trust, a certification-based indirect recommendation, and the reputation are used to compute the trust of a vehicle. An implementation scenario is also presented to analyse the communication delay between vehicles though this is not a real-world scenario. Kerrache et al [125] achieve reliable data delivery and present an intrusion detection module to thwart Denial of Service (DoS) attacks. This scheme calculates the trust of neighbours using the sender's direct trust data, indirect trust / opinion received from the previous relayer, weight of official vehicle, and prior verified sender data. In this approach, packets are forwarded along the most trusted path using a routing protocol. However, this approach only suggests a trusted route for packet forwarding besides thwarting DoS attacks.

In Dahiya et al [10], the trust model employs a false message detection scheme to generate feedback on the received message which is used with the reputation to compute the trust. Vehicles utilize primary and secondary scores from the RSUs for further communication until the next periodic update. This scheme is evaluated in the presence of false messages for both urban and highway environments. Nevertheless, this model suffers from excessive trust metric dissemination. In Li et al [92], neighbours

calculate reputation from past interactions which are stored in a remote centralized server. A vehicle attaches its digitally signed reputation when it sends a message. A receiver accepts the message if it finds the reputation value of the sender is greater than a specific threshold. The authenticity of the sender is verified later, and its reputation is updated on the server. However, in this model, the centralized server is contacted frequently for reputation requests and replies. In Li et al [93], a Reputation-Based Global Trust Management (RGTE) scheme based on Reputation Management Center (RMC) is presented. The RMC keeps track of the updated reputation of all vehicles in the VANET. Every vehicle sends its recommendation about its neighbours to the RMC and then uses statistical central limit theory to exclude unreasonable recommendations. It assigns a new reputation score to vehicles for which it has received recommendations. Whenever a receiver receives a message, it directly consults the RMC about the trust of the sender. However, this model also suffers from the same problems (Li et al [92]). Conversely, in Dotzer et al [117], the researchers present a reputation system based on opinion piggybacking for VANETs. This approach calculates confidence of a received event report using majority voting. Furthermore, it uses direct, indirect and a combination of both trust mechanisms if both trust metrics are available. However, the approach is susceptible to collusion attack when the longest-serving attacker manipulates the reputation of vehicles.

In Atwa et al [126], the researchers use fog nodes to collect and filter the trust records. This approach classifies vehicles as either frequent or occasional visitors. This approach isolates the reputation of safety-related tasks from the non-safety tasks. It is simulated in MATLAB and compares the message overhead against an experience-based trust model. The results suggest that the model reduces the message overhead and offloads the computations to the infrastructure from the vehicles. However, the fog nodes and RSU need considerable communication to obtain the updated trust. In Abassi et al [127], the authors manage trust for a cluster using the reputations of users. First, vehicles form clusters through a VANET Grouping Algorithm (VGA) under a specific cluster head. Vehicles exchange reputation messages inside the cluster. Only the cluster head updates reputation in the reputation table by checking location closeness, timestamp, and forwarding count. If appropriate, it suggests vehicles for blacklisting to the RSU. Also, a formal proof ensures soundness, completeness, and validation through an inference system. However, this approach is not compared against an existing approach, so it is difficult to assess its performance. In Awan et al [128], an RSU is solely responsible for the trust computation of the vehicles, and it collects the recommendation / feedback from vehicles. Besides this, the RSU creates, manages, and merges clusters for the VANET. A new vehicle is assigned to an existing cluster upon a joining request if a cluster is available, otherwise, a new cluster is formed using the vehicle. This approach divides trust into several components, for example, knowledge, reputation, and experience to make it robust against sybil and wormhole attacks. The RSU also identifies malicious vehicles, blocks their communication and prevents joining another cluster. Though they propose a trustworthy cluster,

it requires excessive collection of trust metrics, dissemination, and clustering management at the RSUs which demands significant computational resources.

In reference Tangade et al [9], the researchers use HMAC and digital signatures to manage the trust of vehicles. RSUs evaluate the vehicle trust based on neighbour trust values and rewards. The scheme also measures the communication overhead for the number of vehicles. The verification can use ID-based and batch signatures and the trust computation can also determine punishments. The researchers in Tangade et al [7] also propose an ID authentication and symmetric HMAC-based trust management approach. The receiver vehicles verify the trust of a sender vehicle every time a message arrives. However, they do not differentiate trust verification of official vehicles. Furthermore, the scheme is not validated on any simulator. Reference Tangade et al [129] combines a reward-based trust scheme with hybrid cryptography to secure the VANET. The TA sets the trust of newly registered vehicles, and it updates the trust of the sender vehicles centrally for every message broadcast. The receiving vehicles forward messages to the RSU to verify their authenticity and the integrity of the trust of sender and then check the trust threshold to accept or reject them. This approach assigns different reward points according to the severity of the event based on how it affects human lives. However, the approach does not provide detection of false alarms. The researchers have conducted a theoretical analysis and evaluated its efficacy against a baseline. However, the trust verification of each message initiates communication among the entities from various levels in the hierarchy of core architecture (first RSU to Agent of Trusted Authority (ATA), then ATA to Regional Transport Office (RTO)). This results in considerable communication overhead and adds latency to the decision-making process.

In contrast, Wei et al [8] presents a Bayesian rule-based direct trust and Dempster-Shafer Theorem (DST)-based recommendation trust model. This model combines many independent beliefs about a vehicle to determine its overall trust. However, an incorrect recommendation can falsely influence the trust calculation. The researchers in Najafi et al [114] use a Bayesian filter and watchdog method to compute the trust of vehicles. Vehicles contact an RSU to find the indirect trust of the target vehicle. After that, a vehicle finds the final trust from the weighted direct and indirect trust. Based on this score, vehicles are assigned a state from: malicious, heavily suspicious, lightly suspicious, and normal. Although they validate their scheme against a baseline considering accuracy and errors, they do not include a punishment mechanism. Conversely, the authors in Soleymani et al [130] apply fuzzy logic to calculate the trust using experience, plausibility, and location accuracy. This approach determines location accuracy using fog nodes. It can detect bogus attacks and message alteration attacks. However, vehicles consulting with fog nodes for location accuracy raise the communication overhead. Reference Guleng et al [15] considers fuzzy logic-based direct trust and Q-learning-based indirect trust. This approach analyses precision and recall metrics with varying numbers of malicious vehicles. However, the overhead is high as it involves repeated sensing of messages from neighbours. The authors in Malhi et al [105] also use fuzzy logic and calculate the relaying trust and coordinating trust. Then the final

trust is computed from these two and a trusted path is searched using a set of rules and experiences. However, this model only considers trust-based routing to deliver a message along the most trusted path.

In Zhang et al [109], the researchers propose a software-defined Trust Based Deep Reinforcement Learning Framework (TDRL-RP) for VANETs. This model evaluates trust based on the packet forwarding behaviour of neighbours and the Software Defined Networking (SDN) agent selects the highest trusted routing path using a Convolutional Neural Network (CNN). In this approach, the VANET routing selection problem is modelled as a Deep Reinforcement Learning (DRL) problem where the goal is to find the most trusted routing path from the source to the destination. A deep Q-network followed by a CNN model is used which takes the state as the input for training the network and produces a Q value as the output. This approach is simulated in OPNET (Chang et al [131]) by using TensorFlow and compared against the AODV routing protocol and an existing SDN-based approach. The results suggest that the approach shows a better packet forwarding rate and network throughput. However, the trust model of this approach is used only in selecting a route to a destination.

In Li et al [98], the authors propose an active detection and blockchain-based trust model for VANETs. Vehicles evaluate neighbours' direct trust by sending probes to forward them to an RSU and wait for the acknowledgement from the RSU. Vehicles also collect reference trust of other vehicles with normal behaviours from the neighbourhood to transmit data using only highly trusted vehicles. After this, the vehicle sends an updated trust list to the RSU to upload it into a blockchain. The RSU disseminates trust information to vehicles and pedestrians. This approach is analysed to find the accuracy and error rate and evaluate it against a centralized and decentralized approach. However, the active detection process requires considerable probe packet generation. Mrabet et al [95] presents a decentralized reputation management approach based on Secure Multiparty Computation (SMC) and blockchain for VANETs. In SMC, every member inputs a private number to the shared agreed function and then expects output from the function. In this approach, individual ratings from vehicles are kept secret and the resultant reputation data is made public. The blockchain ensures network-wide consistent reputation data for vehicles. The approach is implemented, and an analysis is also carried out considering collusion attacks; however, it is not compared with any existing approach. Reference Haddaji et al [99] uses multi-level blockchain for trust management in VANETs. This approach consists of three parts; the first one applies a horizontal trust management mechanism at every vehicle using support vector, k nearest neighbour, or random forest. Then vehicles inform classification decision (normal or malicious neighbours) to RSUs. The vertical trust management scheme applies a verification algorithm at the RSUs to deliver the trust list. Third, a Distributed Trust Management (DTM) mechanism allows the RSUs to use the blockchain to share the trust list and to find the class of a vehicle. This model is simulated in Veins and can detect sybil attacks using the VeReMi (van der Heijden et al

[132]) dataset. However, trust formation, aggregation, composition, propagation, and updating require collaboration among various levels.

Luo et al [100] present another blockchain-enabled distributed trust-based location privacy protection scheme for VANETs. Dirichlet distribution is used for trust evaluation between the requesting and cooperative vehicles. After an evaluation, the result is sent to a nearby RSU to form a block to insert into the blockchain. The RSU broadcasts this result to neighbour vehicles and shares it with other RSUs in the network, so the information remains authentic. A distributed k-anonymity method is used to construct a trust-based anonymous cloaking region. The trust of the vehicles is determined by considering the historical trust records and the current behaviours. This approach is simulated in JAVA and the blockchain is deployed in HyperLeader. The security analysis confirms the approach is resilient to bad-mouthing attacks, on-off attacks, whitewashing, and sybil attacks. Additionally, this is compared with a baseline approach and shows the privacy leakage rate decreases gradually over time. However, the malicious vehicle detection rates require at least 20-30 rounds to obtain high accuracy although they treat this as the initialization phase. In Li et al [94], a blockchain-based trust model is also presented to protect the privacy of VANETs. Vehicles use the certificate as a pseudonym to access a Location-Based Service (LBS) while protecting the privacy of a real user. A Dirichlet distribution-based trust management algorithm is also developed to evaluate the behaviour of vehicles whilst the CA manages trust records from users securely into a blockchain centrally. An RSU-dominant algorithm is also developed to construct the cloaking region for protecting privacy. This approach performs a security analysis and trust evolution in the presence of on-off attacks and bad-mouthing attacks. This is a better privacy protection approach than Luo et al [100], but it suffers from latency.

2.8.2 Data-Oriented Trust Models

Reference Gurung et al [133] presents a data-oriented trust model based on content similarity, content conflict, and route similarity. Nevertheless, monitoring of activities is limited in this approach as there is no infrastructure. A weighted voting-based trust model is proposed for VANETs in Huang et al [91]. This model assigns a lower weight to distant vehicles than to closer ones to an event location while vehicles evaluate an event. However, they consider distance as the only metric for trust management. In Wei et al [16], the trust model decides adaptively on received events. Vehicles forward a message based on a predefined number of messages received or the delay between the first message about an event and the current message received from the RSU. This approach is tested using ns-2 (https://www.isi.edu/nsnam/ns/index.html [122]) considering bogus messages, message drop, and message alteration attacks. This is also compared against a beacon-based trust management model. However, the forwarding of events is delayed as a condition needs to be met first.

In Wei et al [113], the researchers propose a data-centric trust model for VANETs. An RSU finds the direct trust of the event by finding the similarity between the beacon and the received event and then informs nearby vehicles. When indirect trust is available, both direct and indirect trust are combined using DST. This approach is evaluated in the presence of alteration and bogus message attacks to evaluate the accuracy, f-measure, precision, and recall compared with two existing approaches. However, this model suffers from delay whilst the RSU decides and shares its opinion concerning an event. The researchers in Basheer et al [134] merge beacons with alert messages to determine the status of vehicles. Receiver vehicles check the stored information about the source and the RSU information whenever they enter a coverage area. This approach divides a digital map into multiple segments and defines the segments from which a vehicle can receive or reject a message. To retransmit a message, this approach checks the predefined number of endorsements from connected vehicles. However, this model does not conduct a security analysis.

In Shaikh et al [111], receiver vehicles verify the time and location closeness to establish the reliability and the message freshness. Then receiver vehicles determine the confidence in the received message from each sender about an event. Based on a threshold trust score, a receiver accepts or rejects a message. However, this approach is reliable whenever data is received from direct neighbours, otherwise, it may suffer from a high false positive rate. The researchers in Wan et al [135] present an approach that evaluates the trustworthiness of the message. This approach considers one-hop Emergency Warning Messages (EWM) and multi-hop Event Reporting Messages (ERM). Receiver vehicles collect messages from in-front vehicles and those vehicles which only pass the event location. Vehicles decide about a received event based on the location and time closeness of the event and whether it comes from the leading vehicle, or if it is driving through the region later. Then receiver vehicles start a timer and upon expiration, it compares the sum of all positive with the sum of negative event reports to trust an event. This approach is effective against bad mouthing attacks, on-off attacks, and sybil attacks. However, this scheme cannot track a vehicle as no identity is included when dispute arises.

The researchers in Rawat et al [136] propose a probabilistic approach to estimate the trust of received information. This approach determines the distance and geolocation from the Received Signal Strength (RSS) of the messages. It does not forward a trustworthy message if it is beyond a specific distance. A Bayesian Inference based voting mechanism called BIBRM assigns a time parameter to each road segment and messages carry the road ID when they are forwarded (Wang et al [137]). Every vehicle has a local map and a database to update these records and recomputes a new route using the Dijkstra algorithm upon reception of a message. However, this requires frequent maintenance of a local database. Reference Rostamzadeh et al [20] proposes the FACT framework consisting of two modules for achieving reliable information dissemination. The first module checks the security of the message and assigns a trust value to each road segment and neighbourhood. If a message is trustworthy, then the

second module finds a highly trusted path for forwarding the message. However, this application-oriented scheme does not allow monitoring from the authority.

In Wu et al [112], the authors use an ant colony optimization algorithm as well as integrating direct observations with feedback information to compute the trust of data. However, this approach introduces longer delays as it needs to collect, analysis and disseminate data. There are some factors associated with this latency which are distance between the nearby RSUs, traffic densities, traffic situation, and evidence member threshold. These factors affect data delivery latency varying from 0 to 37 seconds based on their experimental scenarios. Also, the method used to transmit data between RSUs, and vehicles affects delay. The researchers in Chen et al [138] use DST to combine data from multiple neighbours about an emergency event. In this approach, vehicles verify the location of an event using beacon messages. In their simulation, they consider message alteration, suppression, and bogus attacks to evaluate the reliability of the approach. However, this approach also suffers from delay since it collects messages from neighbours. Another DST-based approach described in Sharma et al [139] utilizes old beliefs from neighbours which may produce false results because of using incorrect trust metrics from neighbours.

Yang et al [101] presents a Blockchain-Based Traffic Event and Trust Verification (BTEV) framework. This framework manages trust, privacy, and security through a two-stage verification of events and a two-phase transaction for fast notification of events. The scheme can thwart selfish behaviour and false message rebroadcasting. However, the two-phase Proof of Event (PoE) consensus produces delays when traffic densities are high and when vehicles request bulk messages that may congest the VANET. Alternatively, Xie et al [102] proposes a semi-centralized trust model which combines Proof of Work and Proof of Stake-based blockchain where vehicles upload trusted traffic information for VANETs. The approach supports both messages and videos about road conditions. Vehicles upload recorded encrypted traffic videos including a message digest and tag to a cloud server to share with other vehicles. Other vehicles score the tag and the RSU calculates trust from the distance between the scoring vehicle and the tag sender. This approach requires 5G base stations and a 5G-enabled SIM (Peterson et al [140]) for each vehicle which operators save in their database. The message is verified first to calculate the trustworthiness of an event and the responsible entity uploads the video into the cloud with blockchain. Malicious traffic broadcasts are deleted, and malicious vehicles are banned. This approach is validated in OMNeT++ (Varga et al [18]) using the crypto++ (https://cryptopp.com [141]) library and a security analysis is performed in the presence of malicious vehicles and an RSU. The trust can be determined accurately in the presence of low malicious rates but when it is over 5%, disparities arise.

In Sharma et al [106], the researchers present a machine learning (KNN, decision tree, naïve Bayes, and random forest) based trust model to detect location spoofing attacks from consecutive BSMs.

Vehicles send BSMs to an RSU which runs a detection framework using the stored data and the received BSMs. The model is trained with a VeReMi dataset (van der Heijden et al [132]) which contains both legitimate and malicious data that helps to classify future information. Their analysis examines the accuracy, precision, and recall for each ML approach. Results confirm that the KNN and random forest show better accuracy in classifying attacks. However, this only analyses BSM data in regard to detecting false position attacks. Reference Mankodiya et al [107] proposes an Explainable AI (XAI) system for trust management of autonomous vehicles using an ensemble learning algorithm and a decision tree-based model to differentiate malicious vehicles from benevolent ones. Explainable AI makes the complex ML and deep learning approach more comprehendible. The simulation of this approach confirms the suitability of this model for VANETs in terms of accuracy, precision, and recall. However, this approach only considers fake positional data to classify malicious vehicles using the VeReMi (van der Heijden et al [132]) dataset. Guo et al [110] proposes a model to evaluate trust with high accuracy in the presence of a high ratio of malicious vehicles. The approach evaluates an event using the external information as well as the sensed and self-experience (internal information) relating to the event using a coefficient-based weighted mechanism. The final trust is then compared against a threshold to determine whether the event is bogus or genuine. Also, a reinforcement model is used to adjust the trust evaluation function based on the previous results. Vehicles which comply with the protocol are normal and those that go against the protocol either intentionally (malicious) or unintentionally (faulty) are considered adversaries. The model is examined in Veins using real-world map data and the precision is compared against voting-based, Bayesian, and DST-based approaches by changing the influence of the false information. For an unexplored road condition, when there is no internal information available at that time, if there is more malicious information then the approach may make a biased decision about an event. As this framework demands the collection of external information to compute the trust of an event, it may delay the computation process until a fixed period is elapsed or a certain number of reports have been received.

2.8.3 Hybrid Trust Models

The researchers in Ahmad et al [11] evaluate one data, one entity, and one hybrid trust model under various adversary scenarios. In addition, an asset-based threat model and an ISO-27005 (qualitative) based risk assessment model are presented for the identification of critical vulnerabilities. In Mármol et al [115], the researchers present an infrastructure-based scheme called TRIP which considers the severity level of the safety messages. Moreover, this approach computes the trust of the sender using direct interaction and indirect recommendations. This approach is simple, fast, accurate, scalable, and resilient to some threats. However, the approach lacks an estimation for communication overhead, privacy, and identity management. In Ahmed et al [142], receivers employ a logistic regression-based trust algorithm to detect misbehaviours in the VANET. The logistic regression iteratively evolves based on the trust score to correct for misjudgements. The trust of the sender vehicle is updated using the

received messages initiated from them. In some cases, when an observation is not present, the receivers utilize learned events from trusted sources to confirm the received message and assess the trust of the sender. The receiver forwards a list of malicious and honest vehicles to neighbours. This model is validated in OMNeT++ (Varga et al [18]) using a circular route to reroute vehicles iteratively. First, they analyse the trust evolution in the presence of on-off attacks. The approach is also compared with a weighted voting and a majority voting approach and measures accuracy to show the efficacy of identifying malicious vehicles. However, a malicious vehicle can take advantage by changing an attribute, for example, brake, acceleration, or location data in the Basic Safety Messages (BSMs). Minhas et al [143] combines role-based, experience-based, and majority-based trust to compute the final trust score. Whenever a receiver receives a message, it consults a local matrix ordered by role and experience to determine the trustworthiness of the message. Neighbours are stored and ordered based on their contributions to form the opinion. However, this model assumes frequent meetings between vehicles which is unrealistic for VANETs.

In Yao et al [144], the researchers present a dynamic hybrid trust model which assigns a weight based on the type of application and the role of the vehicle. The dynamic entity-centric trust model thwarts black-hole attacks and selective forwarding attacks by marginally sacrificing the performance of a Greedy Perimeter Stateless Routing (GPSR) protocol. The data-centric trust model discovers relations among data and evaluates trust based on traffic patterns and utility theory. This work demonstrates the impact of a trust model on a routing protocol and compares this with a GPSR routing protocol. The data trust model can be further improved by selecting appropriate utility parameters. Atwa et al [145] consider the likelihood and impact of taking a decision when both the event and the opposite event coexist. This approach is compared with a multi-facet-based trust model. The result suggests that this approach always selects a low-risk action relative to a typical trust-based approach. However, the model is designed for a clustered environment. In Gao et al [146], the researchers devise a Bayesian inference-based direct and recommendation-based trust model for VANETs. The direct trust calculation considers penalties and time-decaying information. Also, the confidence of direct trust is checked against a threshold to avoid a costly recommendation trust calculation. The approach achieves more successful interactions than the two other approaches. However, the analysis only considers packet drop and interception as malicious behaviours.

In Abdelaziz et al [147], the receiver vehicle multiplies the opinion of the immediate forwarder with its current trust to derive the receiver opinion about the message and decide whether it forwards the message further or not. In this approach, a trustworthy message can be dropped when the trust and opinion of the immediate forwarder are low. In Rehman et al [148], a receiver vehicle both checks the trust status of the sender vehicle and the validity of the event. Also, receivers compare this status with the status from the neighbour vehicles. After that, a receiver compares their opinion with the opinion of neighbours to conclude the trust of the sender, for example, if the opinion matches, then the sender is

trusted, otherwise, distrusted. Neighbours also exchange honest and malicious node lists to update each other. This approach is validated in MATLAB considering congested scenarios and performance comparison shows the effectiveness of the trust computational error and end-to-end delay with varying vehicle density. However, the analysis does not consider any adversary model. The researchers in Mühlbauer et al [14] manage the reputation of vehicles at the RSUs, and vehicles periodically transmit all the noticed events to nearby RSUs. The RSU then announces updated reputations to every vehicle using the recorded event list. Receivers store all the messages about an event in a decision table until a timer expires. Then receivers evaluate an event either true or false. However, this approach suffers from high response time and communication overhead. Additionally, the scheme can experience data coherency whilst vehicles are waiting for updated reputation data.

In Pham et al [149], an adaptive trust-privacy framework consisting of an Adaptive Link-ability Recognition Scheme (ALRS) and an Adaptive Trust Management Scheme (ATMS) is proposed. ALRS preserves privacy by hiding identity and supports trust management by revealing the identity and vice versa. ATMS verifies data from other vehicles and updates reputation. However, the approach suffers from biased decisions when malicious vehicle rates are very high. Reference Zhou et al [116] accepts or rejects a new vehicle based on the direct and indirect trust calculation using a historical security vector of events. The security vector is first calculated based on the vehicle's past behaviours and then the AU is contacted to get the historical security evaluation to finalize the direct trust of the vehicle. In the indirect trust calculation, the vehicle recommendation trust vector is formed from other vehicles. This approach also uses the correlation coefficient to filter out malicious recommendations from benevolent recommendations. However, this approach does not consider any adversary model while measuring efficacy. Conversely, the authors in Li et al [150] first verify the trustworthiness of data and then compute the trustworthiness of a vehicle from the functional and recommendation trust. It considers simple, bad mouth and zigzag attacks, and finds the precision and recall in the presence of truly malicious vehicles to evaluate the accuracy. However, this approach also suffers from delay. Alternatively, in Rai et al [151], a self-organizing hybrid trust model is proposed for both urban and rural scenarios. This approach keeps a history of interactions and then validates the received messages by assigning a credit. This model calculates the trust for each unique message to accept the message with the highest trust for a particular event. This model can detect fake event locations, source locations, and event time as well as can revoke messages from malicious vehicles. However, this model is not evaluated against a baseline.

Gazdar et al [12] checks the reliability of messages considering only direct interaction. Every vehicle stores previous interactions and the trust of all neighbours. This model detects eavesdropped messages and fake events. However, they do not consider false trust messages from malicious vehicles. Liu et al [152] embed the trust certificate of a vehicle with the message that a receiver uses as a weight while evaluating the trust of data. A vehicle that visits the event location sends a status update to confirm or

deny the event. The vehicle sends positive or negative feedback from the local storage to the RSU when enters its coverage area. Upon reception of the feedback, RSU forwards it to the CA to update the vehicle trust certificate. Later vehicle asks RSU about the updated trust certificate from the CA. Thus, the approach demands additional communication with the RSU to send feedback and receive trust certificates frequently which is affecting the channel availability issue. A theoretical analysis has been conducted to illustrate the robustness of this approach considering the unreal event broadcasting and the unfair trust feedback setting.

Koirala et al [153] select the trust evaluators of other vehicles using a vehicle's daily highest running time. This approach monitors the running time of normal vehicles and keeps track of valid and invalid messages broadcasted from vehicles and updates the trust details accordingly. Trust evaluators keep their database updated by sharing the latest information continuously and vehicles receive updated trust information from them. A message from a sending entity includes a signature, timestamp, and trust details. Receiver vehicles first verify the received message by evaluating a hash of the trust and timestamp and then determine a confidence threshold to accept or reject it. This approach is validated in ns-3 (www.nsnam.org [154]) and a theoretical analysis is also presented to confirm that the approach meets authentication, non-repudiation, integrity, and privacy requirements. However, a malicious vehicle can drive for a longer time to become a trusted evaluator and the analysis does not consider known adversaries in their validation. In Rehman et al [155], a vehicle learns cognitively from the environment and develops contexts around an event to infer trust. It forms a context using an ontology which associates a set of interrelated concepts (for example vehicle, evaluation, event). This framework considers experience, opinion, and role for the trust evaluation. For outlier detection, time, speed, and distance thresholds are checked. Besides finding the trust level for every report, this approach also finds the confidence of the report using their method. The framework is simulated in MATLAB using both rural and urban scenarios and compared against existing frameworks using a confusion matrix. However, malicious vehicles can bypass the outlier-based malicious detection process and can send false messages within the acceptable threshold they set for this model.

Hussain et al [156] propose a social network and email-based trust framework for VANETs. Through this approach, the authors minimize the gap between the entity trust model and the data trust model. However, the approach is not analysed with a real-world traffic scenario. Kerrache et al [157] propose an approach called TVNets which can operate across various traffic densities. This approach computes the trust of neighbours, identifies dishonest vehicles with/without RSUs, and discovers the most trusted path to send a message. This approach is compared with a routing protocol in the presence of a black hole attack. It is also tested without RSUs, employing direct and indirect trust, and has scope for improved robustness from other types of attack.

In Pu et al [96], a blockchain-based decentralized trust management approach is proposed. In this scheme receiver vehicles evaluate the trustworthiness of received messages and then accept or reject them based on a threshold. Also, a receiver computes the trust of the sender vehicle and sends the updated trust periodically to RSU that merges these values to compute the reputation of the sender vehicle. The RSU first finds the reputation of the message sender and then packs these reputation values into a block and then operates as a miner to insert it into the blockchain. This approach is analysed in OMNeT++ (Varga et al [18]) and compared against a decentralized model to show it performs better in detecting malicious vehicles and dropping malicious messages. However, in the presence of a high number of malicious vehicles, they can assign higher trust scores to themselves so that the malicious message detection rate may fall. In Yang et al [103], the authors propose a hybrid, three-layer blockchain-based trust model which employs Dirichlet distribution, reputation regression, and punishment revocation. Blocks are stored in the cloud and a CA manages the key distribution of vehicles and the registration of vehicles and RSUs. They consider simple, slander, and strategic attacks along with both normal and malicious servers and reviewers in their analysis to show the precision and recall rate are better than one existing approach. However, this work does not use any reward scheme to motivate benevolent activities from vehicles.

Chen et al [97] present a decentralized blockchain-based trust model which selects a message evaluator through RSU collaboration. The approach determines a rating for messages, the sender, and the evaluator. Then, they calculate the global trust of a node based on the rating and message quality. They preserve trust data in the blockchain and use a consensus process to insert blocks. They claim that their approach can prevent Sybil, message spoofing, bad-mouthing, and ballot-stuffing attacks. However, this model is not compared against other trust-based models. Yang et al [83] propose a blockchain-based intelligent trust model that validates the received message using Bayesian Inference. Each vehicle generates a rating for every source vehicle which the RSU utilizes to generate a block of trusted vehicles. Then it uploads this block into the trusted blockchain. However, a compromised RSU can construct a fake block from untrusted vehicles and insert it into the trusted blockchain.

In Huang et al [104], the authors present a machine learning and active detection-based trust model to evaluate the credibility of vehicles and events. Active detection helps in finding the indirect trust of neighbours and a Bayesian classifier is used to identify a malicious vehicle. Here a receiver vehicle finds the trust of the sender vehicle by multiplying the direct and indirect trust and compares it with a threshold to accept a message. Blockchain is used to store the trust and certificate of vehicles. It achieves V2V authentication using a smart contract on the blockchain. This approach is implemented in Python and shows the trust score evolution with different indirect trust scores and different sender message accuracies. The approach detects malicious behaviour at a fixed time from specific vehicles. However, each active detection requires two extra messages for every evaluation. In Tangade et al [108] the authors propose a deep learning-based trust computational approach to thwart internal attacks. The deep

neural network consists of four hidden layers and calculates reward based on driver behaviour and classifies honest and malicious behaviour using another deep neural network. The trust of a vehicle is calculated from reward points using a three hidden-layer deep learning approach by the receiver vehicle whereas the message is classed as fraudulent or non-fraudulent by an RSU using a two hidden-layer deep neural network. This approach is implemented in ns-3 (www.nsnam.org [154]) using the open-source library TensorFlow 1.6.0 and then compared against two other approaches considering the computational overhead. However, the approach lacks an efficient trust update mechanism as it requires a chain of communication to the top-level authority of the architecture from the vehicle for every message broadcast.

2.8.4 A Comparison on the State-of-the-Art Trust Models

In this section, trust models from the literature are compared in Table 2-1 to illustrate their differences considering basic principles, trust metrics / feedback dissemination, and trust evaluation. Also, this comparison considers what types of adversaries each trust model can thwart as well as what analysis was used to validate each model. Furthermore, whether an approach is compared with a baseline or not is recorded in this comparison and what traffic scenarios a trust model considers during the validation of their model is also given. However, all of them follow receiver-side evaluation. Hence, they suffer either from communication overhead or from decision latency and some approaches suffer from both. The reason for this is explained in Section 1.2.

Table 2-1: A Comparison of Trust Models

Ref.	Type	Any Roadside units?	Underlying Principle	Feedback Collection	Trust computation	Adversary Model	Analysis	Baseline Comparison	Traffic Scenarios	Simulators
Dahiya et al [10]	Entity	RSU	The scheme evaluates the accuracy of information from feedbacks and removes malicious feedback. Blacklists vehicles.	It generates feedback	Each vehicle gets a score based on the accuracy of information.	False message	An analysis to show the deviation between the actual and estimated accuracy for different scenarios	No baseline considered	Urban and highway	Veins, OMNeT++, and SUMO
Haddadou et al [13]	Entity	No RSU	TPM of vehicles has credit score to use for sending or receiving messages. To send a message, the transmission cost is determined, and it is paid back when a sender sends a true event.	Each receiver sends acceptance or refusal notification to the sender.	Majority opinion from the feedback data used to find the reward value. A reward is added to the credit account of sender vehicle.	Alteration attack, false information, selfish behaviour	Malicious vehicle detection delay and false positive rate. Reception rate of corrupted data and reception ratio in presence of selfish nodes.	Node detection percentages (in presence of varying rate of malicious vehicles) are compared theoretically.	Urban and highway	ns-2, VanetMobiSim, and SUMO
Guleng et al [15]	Entity	No RSU	Fuzzy logic considers cooperativeness, honestness, and responsibility to find direct trust. Q learning evaluates the trust of non-neighbour node.	Repeated sensing of "hello"	Trust is calculated from the direct and indirect trust.	Bad mouth attack	Analysis of precision and recall. Packet delivery ratio under varying numbers of malicious vehicles.	It is compared against a deterministic trust and a scheme which does not employ trust.	Freeway has two lanes in each direction	ns-2
Li et al [92]	Entity	Access points	The reputation score determines whether an event from a sender should be reliable or not. The reputation server revokes the reputation of malicious vehicle when specific condition holds and does not provide any certificate.	Feedback	When the time decaying reputation score of the sender vehicle is greater than a threshold, the receiver vehicle accepts the message from the sender.	Theoretical proof against false message and reputation manipulation attacks.	Analysis of message drop rate versus access point distribution and unavailability of access point and reputation server.	compared with two existing approaches	Urban scenario taken from Pittsburgh	GrooveNet

Mrabet et al [95]	Entity	RSU	Infrastructure computes global reputation, verifies blocks, and insert into the blockchain. CA manages access to the network.	Collects feedback from the regular vehicles.	Reputation is calculated as the average of all ratings.	Collusion attack	Some performance tests. Reputation calculation Update into the blockchain	No baseline comparison	No	Ethereum test blockchain Ganache
Pu et al [96]	Entity	RSU	RSU uses multicriteria decision to compute reputation. It packs the list of reputations into a block to insert into a blockchain.	RSU collects trust values of event sender.	RSU calculates the reputation value of the message sender from the trust values of validator vehicles.	No adversary model.	False message detection rate, malicious vehicle detection latency. Number of dropped false messages, and average trust value.	False message detection rate is compared with a baseline.	Manhattan grid	OMNeT++
Li et al [98]	Entity	RSU	After the data transmission, a vehicle sends updated trust to the RSU to verify and update into the blockchain. Active detection based malicious vehicle detection from the surrounding.	Probe message, and updated trust from neighbours.	Vehicle trust is calculated from the detection trust, reference trust, and transmission trust.	Packet drop, spoofing, and cooperation attacks.	Analyse delivery ratio, detection ratio, and average trust value under different ratio of malicious vehicles.	Compared with two baselines in terms of delivery ratio, detection ratio, and average trust value.	Vehicles and pedestrians are randomly distributed in a square area of(5000m X 5000m).	ns-3
Haddaji et al [99]	Entity	RSU	Horizontal trust scheme detects malicious vehicle using different machine learning algorithms. The vertical trust scheme verifies trust. A blockchain contains the vehicular trust list.	Many votes/decisions are collected about a vehicle.	RSU sums all the votes for a node and check if it is greater than a threshold to put into a trusted list.	Sybil attack	Accuracy versus amount of collected data using SVM, Random Forest, and KNN	No baseline mentioned	Traffic scenarios are not mentioned	Ethereum environment
Luo et al [100]	Entity	RSU	Dirichlet distribution-based trust management approach and blockchain is used to store trust of vehicles. Distributed k-anonymity based cloaking region maintains the privacy of vehicles.	Recorded historical trust information is queried to update the trust of other party.	Historical trust information of vehicle is added with the trust degree as a requestor and cooperator.	Bad mouth and on-off attacks	Theoretical security analysis of the mentioned attacks. An analysis of malicious vehicle detection in presence of on-off attack	Probability of location data leakage and percentage of malicious vehicles in the cloaking region are compared with two other schemes.	Real driving data collected during 24 hours of driving in Cologne, Germany.	HyperLeader, JAVA
Yang et al [103]	Entity	RSU	Trust model uses Dirichlet regression, and punishment mechanism. Blockchain stores the rating of service providers.	Service rating of service provider.	Trust evaluation considers both positive, neutral, and negative ratings. Malicious reviewer and servers are blocked when threshold condition meets.	Simple, slander, and strategic attacks	Analysis of reputation of normal or malicious vehicles. An analysis of collaborative vehicles and success rates versus epoch. Find the FP and FN vehicles.	This model is compared with beta distribution-based trust model.	24-hour taxi GPS data collected from Chongqing.	Not mentioned
Mármol et al [115]	Entity	RSU and base stations.	Trust and reputation approach which isolates malicious users from the network. Each vehicle computes trust of other vehicle from which it receives a message.	Recommendation.	Trust is computed using the reputation, recommendations from both neighbouring vehicles and RSU. When trust is determined three fuzzy sets decides on the received message.	False message, and collusion attacks.	Accuracy and scalability evaluation with and/or without collusion attack.	Not compared	Fixed traffic mobility in an area of (100mX 100m)	Bespoke simulator, TRMSIM-V2V
Zhou et al [116]	Entity	Fixed access point	Secure authentication based direct trust and indirect trust calculation scheme. Correlation coefficient is used to filter out malicious recommendation.	Application data.	Trust is calculated from the direct and indirect trust. Indirect trust is calculated from the average of all recommendations collected by the Authority Unit(AU).	On-off attack and bad-mouthing attack.	Security degree analysis. Indirect trust evolution.	No baseline Comparison	Network scenario is not stated.	MATLAB
Saraswat et al [124]	Entity	No RSU	Analytical Hierarchy Process (AHP) based method which utilize direct and indirect trust.	Recommendation.	Previous reputation is added with the direct and indirect trust to get the final trust.	No adversary model.	Analysis of communication delay in presence different number of vehicles.	No baseline comparison.	A road with multiple unidirectional lanes.	Not mentioned
Atwa et al [126]	Entity	Fog network, RSU	Task-based Experience Reputation (TER). Fog nodes collaborate to collect and send aggregated trust to RSU. Differences of frequent versus occasional visitors.	Not required.	Accumulate reputation based on task basis reward and punishment.	No adversary model.	Analysis of communication overhead in presence of well-known and experience-based trust model. Workload analysis of both	Compared with experience-based trust.	Random distribution of vehicles.	MATLAB

							experience and task-based model.			
Uma et al [121]	Entity	RSU	This approach has a trust manager, route manager, and decision manager.	Neighbour information.	Trust is computed from past experiences, neighbouring vehicle information, trust of the vehicle, and the packet delivery ratio.	No adversary model.	Analysis of packet delivery ratio and delay.	Compared with baseline in terms of packet delivery ratio.	A grid topology is used	ns-2
Zhang et al [123]	Entity	RSU	Bayesian inference based local trust calculation of vehicles. TrustRank based algorithm calculates the global trust of vehicles. Trust is propagated through seed vehicles to other vehicles which is like a Markov process.	Local trust.	Local trust is calculated using Bayesian distribution. Global trust is computed using iterative formula and next decaying factor is applied on it.	Newcomer, collusion, bad-mouthing, and on-off attacks.	Trust evolution in presence of different ratio of malicious vehicles. Analysis of true positive rate and true negative rate.	Compared with baselines.	Motorway in Beijing	Veins, SUMO, OMNeT++
Kerrache et al [125]	Entity	No RSU	It finds the trusted routing path using the link quality and trust. It filters out malicious data.	Not required.	It calculates the trust of neighbours using the role-based trust, recommendation trust and historical trust.	DDos attack.	Analyse the inserted traffic from various nodes to detect DDos attack, selfish behaviour.	Compared with two other schemes.	10 Km highway, with two lanes in both directions.	ns-2, VanetMobiSim
Awan et al [128]	Entity	RSU	Trust based clustering scheme. RSU chose cluster head centrally. RSU blocks malicious vehicle access in the network.	Trust propagation.	RSU calculates the degree of trust using knowledge, experience, and reputation. RSU takes the mean of old trust and new trust to assign new trust to vehicle.	Wormhole and Sybil attacks.	Analysis of average cluster duration, average cluster head lifetime, control overhead, and throughput. Wormhole and sybil attack detection.	Compared with two baselines	Map of Islamabad	OMNeT++, SUMO
Tangade et al [129]	Entity	RSU	A V2V authentication and trust evaluation scheme. ATA evaluates the trust value of vehicles based on reward points.	Safety Message verification and acknowledgement message.	Trust is updated by adding the current trust with the reward point or punishment factor for the current safety message.	Theoretical robustness against impersonation, repudiation, and message tampering, and identity disclosure attacks.	Computation and communication overhead under different traffic densities.	Communication overhead is compared with existing algorithms.	A two-lane two-way highway.	ns-3, SUMO, MOVE
Soleymani et al [130]	Entity	Fog nodes	Fuzzy logic-based trust model considers the message lifetime, previous interaction with the sender, fog node opinion about the event. Relays a message if the sender is trusted.	Event confirmation from the fog node.	Finds the trust of the sender vehicle from the fuzzy logic approach.	False message, message alteration attacks.	Accuracy evaluation	No baseline	2km X 2km	ns-2, SUMO, MOVE
Gazdar et al [12]	Hybrid	No RSU	A distributed tier-based message dissemination scheme. Computes the trust of vehicles which disseminate traffic events. Classify trusted and malicious vehicles.	Not required.	Receiver vehicle checks the authenticity of the message and assigns trust to the senders.	Fake message and message alteration attacks.	Analysis of expected state and probability of being at a state.	No baseline.	Urban area (4000m X4000m)	Not mentioned
Rostamzadeh et al [20]	Data	No RSU	An application-oriented scheme. It checks the message is trusted and then assigns a trust value to each neighbourhood road segment. Each message is transferred through the safest path towards the destination	Not required.	The path trust is calculated from multiple dimensions for example, delay, reliability, security, privacy, and anonymity.	Theoritical proof against false message , message alteration, relaying to another path.	Packet delivery ratio versus delay and delay versus speed.	Compared with two baselines.	Highway	MATLAB
Huang et al [91]	Data	No RSU	Weighted voting-based trust model. The vehicle that is closer to the event has higher weight.	Vehicle receives opinion from the in front vehicles.	A vehicle decides about an event using a sum of weighted opinions from in-front neighbours.	Selfish behaviour.	Percentage of incorrect messages are compared for different voting schemes.	Different voting schemes are examined.	Set of road intersections.	NCTUns, C++
Chen et al [97]	Hybrid	RSU	Blockchain technology based Trusted Execution Environment (TEE) is presented. The lower layer of the hierarchy validates message and blocks whereas the upper layer manages trust, incentives as well as consensus.	Trust credits are made public.	Receiver verifies messages with signed reputation. Each node receives a global trust credit based on the quality of the messages sent and rating results.	Theoretical proof against sybil attack, Message spoofing attack, bad mouthing attack.	Average throughput and response time on message evaluation using TEE and without TEE.	Average latency in their consensus mechanism is compared with two other schemes.	No traffic scenarios mentioned.	A high-level Performance Evaluation Process Algebra (PEPA)
Yang et al [101]	Data	RSU	Both vehicles and RSU needs threshold number of alerts to verify an event. RSU inserts the validated events into a blockchain.	Collect traffic information to verify an event.	Trust of an event is validated through threshold-based validation process.	False attack	Impact of percentage of attackers on the false event success rate,	No baseline mentioned.	Real traffic data collected from vehicle detectors in	ns-3

							Analysis of synchronization time of consensus algorithms.		Taiwan highway.	
Xie et al [102]	Data	RSU and 5G base station.	A vehicle uploads a traffic event video on the server with attached road situation tag which other vehicles score. RSU authenticates and calculates the trust of sender vehicles.	Scores of tag (feedback).	RSU calculates trust of tag using the distance between the sender and the scoring vehicles.	Fake traffic information	Accuracy of malicious vehicle detection, encrypted traffic video overhead, transmission delay versus message rate.	Not compared.	(1000mX 1000m) area, where vehicles move in a random direction.	OMNeT++, crypto++
Sharma et al [106]	Data	RSU	ML approach uses pair of BSMs to detect location spoofing attack. KNN, Random Forest, Decision tree, and Naïve Bayes algorithm are used to detect fake position.	Not required	Two consecutive BSMs from vehicles are analyzed using different ML and then classify the source as legitimate or malicious. Consideration of binary and multiple classifiers.	Location spoofing attack.	Precision, recall, F1-score	Compared with a baseline.	Not considered.	VeReMi dataset, simulator not mentioned.
Tangade et al [108]	Hybrid	RSU	Deep learning-based driver classification scheme.	Not required.	Trust of vehicle is computed from reward points. It classifies fraudulent and non-fraudulent message/driver.	No adversary considered.	Normalised reward points. Computation overhead versus vehicle density.	Compared with baseline schemes.	Not mentioned	Python, ns-3, TensorFlow
Guo et al [110]	Data	RSU	It encompasses data formalization, trust evaluation and strategic module. A reinforcement learning model is used to fine-tune the evaluation strategy.	Information from neighbour.	An event is verified using both external and internal data.	Faulty and false message.	Analysis of precision ratio using different number of rounds.	Compared with other schemes to show proposed scheme outperforms when malicious rate is higher than 50%.	Map of Huangpu district.	OMNeT+, SUMO, Veins
Shaikh et al [111]	Data	No RSU	It verifies the location and time closeness. It has confidence module, trust management module, and decision module.	Not required.	A receiver vehicle determines the confidence on each unique message and trust of each message about an event and then the message with the highest trust is selected from the decision module	False information about location and time.	Effect of malicious nodes on trust and confidence value. Fake location detection analysis. Theoretical proofs against malicious behaviour. False positive rate under various malicious node rate.	Not compared with a baseline.	Suffolk county road map.	SWAN++, ONE simulator
Wu et al [112]	Data	RSU	It employs ant colony optimization which uses both direct observation and feedback to evaluate the trustworthiness of data.	Feedback	Trust is calculated from the correct data, faulty data, and cooperatively falsified data.	Cooperatively-falsified data attack.	Analysis of trust under various observing conditions. Resilience to cooperatively-falsified data attack. Data delivery delay versus distance and densities.	Not compared with a baseline.	Highway	ns-3
Wei et al [113]	Data	RSU	Tanimoto coefficient is used to find the similarity between the event and beacon. RSU can check which messages are more trustworthy than others and disseminates its opinion to neighbours. It is cryptographic and pseudo-identity-based trust scheme. RSU calculates the confidence of opinion using a distance-based method.	Opinion	Trust is calculated from the direct and indirect trust.	Alteration and bogus message attacks.	F-measure on threshold, gamma, malicious vehicle rate and amount of vehicle. Illustration of detection delay versus malicious vehicle rate and amount of vehicle.	Compared with a baseline to show better decision delay.	Random trips on the street map.	ns-2
Dotzer et al [117]	Data	No RSU	Concept of separate event area, decision area and distribution area. Situation recognition. Opinion piggybacking.	Not required.	Overall trust is calculated from the direct and indirect trust. Confidence decision is taken based on different situation reputation constraint.	Modification attacks.	Not mentioned	Not mentioned	Not mentioned	Not mentioned

Wan et al [135]	Data	No RSU	Vehicles ignore messages coming from behind. It considers warning (running state) and traffic events, Also, it assumes an event and opposite event. Forwards a message when it comes from in front and influential area and does not exceed a threshold time.	Not required.	It verifies message location, message generated from in front area of a receiver, and it checks whether a vehicle drives through this region later. The scheme decides based on messages receiving latest time.	False message, collusion, bad mouth, on-off attacks.	Analysis of trip completion time and number of cheated nodes under different malicious node rate, CO2 emission.	Compared the result with different ML approaches.	Map of Nantong city	OMNeT++, SUMO, Veins
Chen et al [138]	Data	No RSU	Tanimoto coefficient is used to crosschecks beacon message with alert message to determine the higher trustworthiness. Messages are encrypted and pseudo-identity are attached while transmitting.	Indirect trust (opinion propagation).	Dempster-Shafer Theorem (DST) is used to combine multiple opinion from neighbours.	Alteration, bogus, and message suppression attacks.	Analysis of F-measure under alteration and bogus attacks. Analysis of location privacy schemes. Detection delay under alteration attacks.	Compared with multiple trust schemes.	Manhattan grid	ns-2
Ahmed et al [142]	Hybrid	No RSU	It first evaluates the received information for correctness using own observations and sender's trust. Finally, classifies vehicles and exchange list of honest and malicious vehicles.	List of honest and malicious nodes.	It finds the trust of sending node using logistic regression. Trust is updated using the received message from sender.	On-off attack	Trust evolution Accuracy of classifying vehicles based on sending events.	Compared with majority voting and weighted voting scheme.	Circular highway.	OMNeT++
Yao et al [144]	Hybrid	No RSU	It has separate entity and data centric trust model to achieve secure routing and improve data delivery rate respectively.	Recommendations	The entity trust is calculated from the direct and recommendation trust. The trustworthiness of data is calculated using utility theory from different factors.	Blackhole attack, selective forwarding attack	Packet delivery ration, end to end delay, path length are evaluated.	Compared with different routing protocols.	(1 Km X 1Km) area where vehicle moves on some selected roads.	VanetMobiSim
Gao et al [146]	Hybrid	No RSU	Penalty and time decaying factors are used to understand better relation. Confidence of direct trust is calculated first.	Recommendation	Node's trust is computed from direct and recommendation trust.	Not mentioned	Analysis of direct trust with successful and failed interaction. Packet delivery ratio versus malicious node rate.	Compared with two baseline schemes.	Manhattan mobile model using 5 X 5 grid.	Not mentioned
Pham et al [149]	Hybrid	No RSU	Adaptive trust and privacy framework. Identity-based cryptographic scheme. Includes both subjective and objective evaluation of event and update sender's reputation.	Opinion	Weighted voting-based calculation is used for subjective trust. Use transitive trust for entity trust. Upon validation and recognition, node updates reputation of another peer.	Brute force and fraudulent message.	Detection rate and correct decision rate versus variable number of attackers.	Compared with one baseline.	Streets on 5Km X 5Km area.	ONE
Rai et al [151]	Hybrid	No RSU	For self-organized VANETs. Can revoke malicious nodes and discard fake messages. Fake source, event time detection. Different credit method used for urban and highway.	Not required.	Receiver evaluates the sender's message to update trust of sender using sender/event location, event time, history of interactions, urban/rural mode and received event. Selects the message with highest trust when trust of each unique message is completed.	False information	Theoretical resilience against false location, event and time spreading. Analysis of travel time, CO2 emission, communication overhead, and accuracy by varying number of malicious nodes.	Urban versus rural scenarios are compared.	Highway and urban (map of Jeddah).	MATLAB, Veins
Liu et al [152]	Hybrid	RSU	It uses entity trust in the data-oriented trust evaluation. Considers distinct messages from different senders about an event. Vehicle revocation. Vehicle collects trust certificate from CA which is included in the messages and extracted as weight for data trust calculation.	Trust feedback reporting.	Vehicles visiting the event location confirms or denies the event. Receiver first verifies the message using some criteria. Then finds the trust of data using a computational method.	Fake message, tampering attacks. Trust manipulation and unfair trust feedback.	Average trust of honest and malicious vehicles. Correct decision percentage of vehicles. Number of real and fake broadcasts for an emergency event	Compared with baseline.	Guangzhou Highway	SUMO
Rehman et al [155]	Hybrid	RSU	It builds context cognitively for an event to trust an event. Time, speed, and distance-based anonymity outlier detection method used.	Opinion	Trust evaluation module uses experience, role, opinion, and thread-based trust.	False message	Accuracy, error rate, precision, recall, and F1-score. Malicious node detection	Trust levels are compared with other schemes.	Both urban and rural.	MATLAB

Hussain et al [156]	Hybrid	RSU	Email and social network-based trust.	Gather sender's trust, two-hop trust propagation allowed.	If receiver finds the sender in the trusted list, then the message is trusted, otherwise, sender's trust is asked from neighbours within 2 hops. Receiver calculates trust using social interactions and intermediate trust.	No adversary model.	No analysis conducted.	No baseline comparison.	No scenario considered.	Not simulated
Kerrache et al [157]	Hybrid	RSU	It is based on direct, indirect, event and RSU-based trust. Revokes dishonest node collaboratively. Notion of inter vehicle trust and RSU to vehicle trust. Collects and combine indirect recommendations about a node.	Positive or negative recommendation.	It evaluates message quality, event effectiveness. Each vehicle send neighbour evaluation to RSU which RSU uses to compute RSU to vehicle trust. Final trust is calculated from vehicle to vehicle and RSU to vehicle trust.	False, DoS, platooning, and message dropping attacks.	Average end-to-end delay versus number of nodes. Packet delivery ratio, throughput. Dishonest vehicle detection ratio.	Compared with baseline schemes.	Valencia city map.	ns-2 and SUMO

2.9 System Requirements

A VANET should enable only trustworthy message dissemination to participants. At the same time, it should dissuade abuse of the system by demotivating and isolating bad drivers at the earliest possible opportunity and simultaneously motivate good drivers. Since road incidents are common we cannot avoid them, but we can work towards minimization of their aftereffects. In this way, drivers may experience traffic comfort and their travel time can be kept within an acceptable range as they may detour around issues in a timely manner through appropriate announcements. Many trust communication models proposed are based on state-of-the-art technologies such as machine learning (Guleng et al [15] and Huang et al [104]), blockchain (Xie et al [102] and Malhi et al [105]) and probabilistic models (Rawat et al [136] and Wang et al [137]). However, they mainly follow receiver-side trust computation of the source and/or its messages which has drawbacks in terms of communication overhead and higher decision latency. Conversely, regulating communication at the sender-side avoids these pitfalls. In this section, the requirements for the proposed sender-side trust management framework are discussed. We then group these requirements based on the part which they need to have with the trust framework. They are developed after evaluating the shortcomings of existing trust models. As we identified, existing trust models suffer from high communication overhead and decision time. We made these our motivation and would like to address by the proposed trust mode. We follow this listing a step-by-step guide to build the proposed trust framework.

2.9.1 Requirements for the Framework

1. The framework should support multiple classes of vehicles. The framework does not need to manage the trust of official vehicles as they constitute the authority.
2. The maximum trust of any regular vehicle should be lower than the trust of official vehicle.
3. The proposed trust model should be able to manage trust at the sender side. Thus, the receiver vehicles can believe messages instantly without further communication.

4. The trust management framework will only allow announcements from the trusted senders, via tiered access control based on a trust score, and block announcements from untrusted sources.
5. Messages are organised into classes within the framework so that the trust of the driver determines the class of message he or she can announce.
6. The trust of a driver cannot be modified by the concerned driver or any other person. A TPD is necessary to update trust securely to prevent malicious modification.
7. As a vehicle can be driven by multiple drivers, the trust model should accommodate the calculation of trust of each driver individually. Also, the TPD should implement blocking access for the malicious driver and not the vehicle.
8. The TPD should run a periodic blocking checker and execute access blocking in the VANET when the blocking decision message comes from the TA.
9. The trust score of a regular vehicle driver should be within the acceptable range of values and if the lower limit is reached, the TPD informs the TA to block access of network for that driver.

2.9.2 Requirements for Rewards / Punishments

1. Reward or punishment for message announcements should be issued inside the TPD and varied according to the promptness of generation and accuracy of the message.
2. The reward level should be varied as per the networking activities for example, this amount will not be the same for an original announcement, relaying, and beaconing.
3. Fixed rewards for trustworthy communication and incremental punishments should be issued to demotivate drivers from the announcement of untrustworthy messages in the VANET. The punishment strategy should consider repeated attacks from a specific driver. A mechanism must exist to limit the number of times untrue messages can be announced. To limit this situation, an arrangement of incremental punishment should be issued to malicious drivers so that they will be blocked earlier when repeated untrue attacks are launched from them. Also, a policy may block a driver earlier from the network if he/she announces three severe untrue events into the VANET. However, this number can be relaxed for less severe event reporting.
4. A beaconing reward and/or relaying reward should not be given (non-announcing, reporting, nor clarifying) when a driver's trust score is greater than or equal to 0.5. This will motivate poorly scoring drivers to communicate actively and earn trust.

2.9.3 Requirements for Untrue Announcement Detection

1. As a highly trusted vehicle may announce an untrue message, a mechanism must exist to enable an RSU to find out which driver broadcasted the untrue message. To enable this functionality, an RSU implements one of the multiple techniques to determine the validity of an event for example methods such as majority voting, or weighted voting.

2. Whenever a dispute is resolving, an RSU should inform the nearby RSUs immediately to avoid the costly invocation of dispute resolution process.
3. An RSU should collect feedback from the trusted clarifiers and official vehicles around the event for which the dispute arises.
4. An RSU should give higher priority to feedback from official vehicle(s) when deciding on the accuracy/validity of an announcement.
5. An RSU should issue fixed levels of reward or punishment to drivers based on their actions. This is separate from TPD rewards and punishments.
6. Some drivers may switch their behaviour from malicious to benevolent by broadcasting both untrue and true messages. These inconsistent behaviours should be detected and punished appropriately.
7. For the advanced version of the proposed framework, the RSU rewards or punishments should be varied based on factors such as: driver history of rewards or punishments, severity of incident, and the RSU confidence in the data from the sender or reporter(s).

2.9.4 Requirements for the RSU-TA communication

1. RSUs should inform the TA of an untrue attack decision as soon as it is available. The TA stores decisions obtained from RSUs into its central driver profile database. This database contains information about each individual driver's past actions in terms of rewards or punishments along with supporting information.
2. When an RSU asks for any data from the TA, the TA is able to provide the requested information.
3. The TA should maintain a list of neighbour RSUs for every RSU in the VANET. This is useful for when it has to send access-blocking/reward/punishment message towards the concerned driver's TPD.
4. The TA should check the access-blocking condition for a driver whenever it receives a punishment decision about him / her from an RSU. This is to determine if blocking is required.
5. The TA should not announce messages for the network instead it should be regarded as a repository of information saving all the incidents up to a configurable period.

2.10 Trends and Issues

In the previous subsection existing trust models for VANETs are reviewed. Vehicles inform neighbour vehicles and nearby RSUs whenever they encounter an event on the road. To the best of our knowledge, the receivers in existing models start computing trust after receiving a message. They do not verify the trust of the sender while it is announcing a message. As there is no limit on message generation, even an unknown vehicle may broadcast a false emergency message and deceive vehicles. Fig. 2-2 shows that a message from vehicle V causes vehicles A, B, C, W, X, Y, and Z to compute trust

individually irrespective of whether V is a trusted or untrusted source. Also, they may individually contact each other to obtain direct and indirect trust as well as a nearby RSU to obtain information of the sender vehicle. This may involve further data collection at an RSU to provide feedback to the asking vehicles. After this, they apply an evaluation algorithm to compute the trust of the sender vehicle to decide whether to accept or reject the message. Thus, these approach-inherent messages result in additional overhead and delay the trust evaluation process which creates a performance bottleneck. This process demands additional resources from the network. A dedicated bandwidth of 5.9 GHz is reserved with the WAVE protocol stack to support the communication in VANETs. In future, we may expect more bandwidth will be reserved for communication. Even so, we should not waste precious bandwidth when managing a single traffic event as there may be concurrent traffic events in a region. The sequence diagram in Fig. 2-3 shows a general sequence of communication in most existing approaches between the three actors: the RSU, sender, and receiver vehicles. We desire an approach that can reduce the utilization of the network resources and latency when an event occurs. Moreover, messages from an untrusted source should be discarded after evaluation. It is therefore better for a network to restrict messages from an untrusted source. With suitable access control, many independent evaluations can be avoided as the likelihood of sending untrue messages from an untrusted sender should be suppressed.

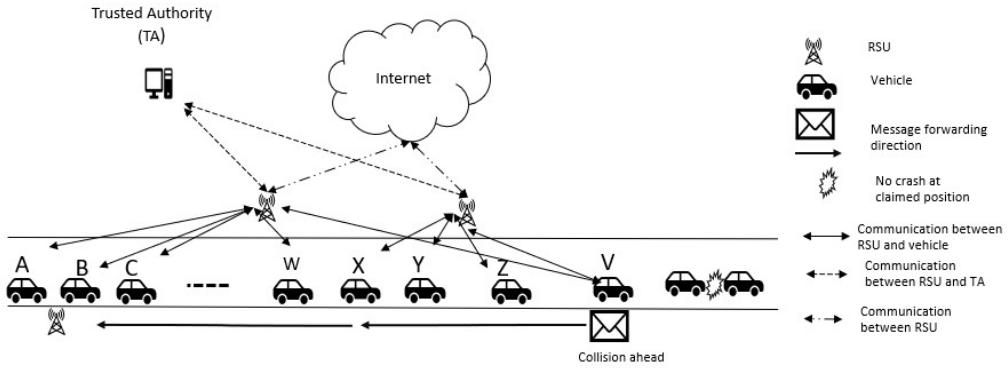


Fig. 2-2: Illustration of an Untrue Message Triggering Trust Computation at All Receivers

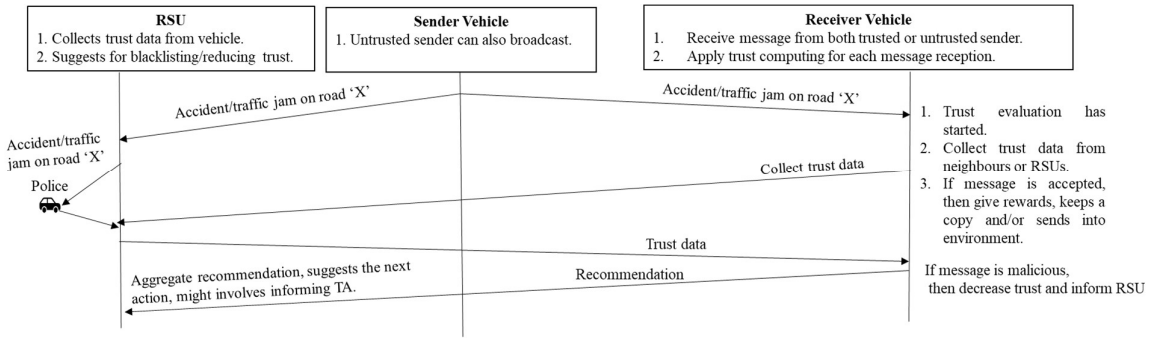


Fig. 2-3: A Sequence Diagram of Event Broadcasting in Existing Trust Models

It is seen in existing approaches that any vehicle can broadcast messages in VANETs as their trust is not evaluated at the time of sending messages, as is the case of vehicle V. Thus, V cannot be considered a reliable source until its trust is evaluated at the receiver vehicles. This situation arises in most existing approaches. They discard messages from an untrusted sender only after consuming lots of resource. Otherwise, they accept a message and treat the sender as trusted. These communications from untrusted sources produce traffic surges in the VANET. Additionally, a false recommendation from an indirect source contaminates the resultant trust computation. Indirect recommendations are also a concern, if relayed via an untrusted vehicle. To identify fake recommendations, some approaches use additional filtering methods, but this adds an extra level of complexity. Furthermore, when vehicles takes higher time to decide on an event that takes them to the hazardous zone which results in more traffic chaos than the reported event. For example, severe traffic congestion or jam would be seen around an accident if the scheme lacks fast driver decision time even the accident event is announced timely. Thus, the delay in decision causes vehicles to enter the hazardous zone. From there, they do not find an escape or detouring option to reach their destination timely. This case is common with the receiver-side based evaluation schemes as they evaluate events after arrival which requires further communication with neighbouring RSUs and vehicles as well as computation. Thus receiver-side trust schemes suffer from higher driver decision time which results more traffic chaos. Contrarily, sender-side evaluation scheme does not need to verify event after arrival. Hence, it offers lower driver decision time compared to receiver-side evaluation scheme. Consequently, vehicles can detour when there is an alternate route to use and avoid the hazardous zone.

There may be some occasions when V is authorized, but its trust is not yet established. A false message from it abuses the network resources. Also, some approaches either collect trust information from the neighbourhood or globally. However, they do not cope well with rapid topological changes. Receiver-side approaches are vulnerable to attacks, performance, and complexity issues. Also, some approaches do not cater for high-priority messages from official vehicles.

Consider the situation when a vehicle receives an accident message from another vehicle. It must decide what to do next. For example, it can choose to avoid the area without waiting for the trust evaluation. Or it can initiate the trust evaluation of the sender and/or its message. The vehicle may then decide on which way it will drive. It may use the original route, or it can select an alternative path. If it acts without awaiting the evaluation, for a false message, it helps the wrongdoer to achieve his/her goal. In most existing approaches, vehicles select the second option which requires time to take a decision. By this point, some vehicles may have entered the event area. This diminishes the impact of an emergency announcement. This delay until a decision is reached is an issue with existing approaches. The slow response time can aggravate a situation. Conversely, in an environment where only trusted drivers can announce a message, receivers do not need to wait for verification of a message. Using

sender-side control based on the trust score, receiver vehicles are no longer required to wait for any decision from other trust entities and can instantly decide what action to take.

Fig. 2-4 illustrates the basic decision-making mechanism followed by the trust-based approaches listed in (Tangade et al [9], Dahiya et al [10], Gazdar et al [12], Rostamzadeh et al [20], Huang et al [91], Malhi et al [105], Chen et al [138], Rai et al [151], and Arshad et al [158]). Here, every receiver individually evaluates the trust of every received message and/or the sender. Besides this, they communicate with the RSU and other neighbours to obtain trust information at run time. The RSU also collects data from official vehicles or trusted vehicles about an event. Alternatively, some approaches do not use any infrastructure (Wei et al [8] and Guleng et al [15]), but they generate sizeable messages among neighbours periodically to manage the trust. For example, the trust approach in Wei et al [8] uses Bayesian statistics to calculate direct trust and uses the Dempster-Shafer theory to determine recommendation trust. Alternatively, the trust model listed in Guleng et al [15] continuously sends and receives hello messages to collect data as well as exchange trust metrics between neighbours. To some extent, these two approaches follow the pattern of interaction shown in Fig. 2-5.

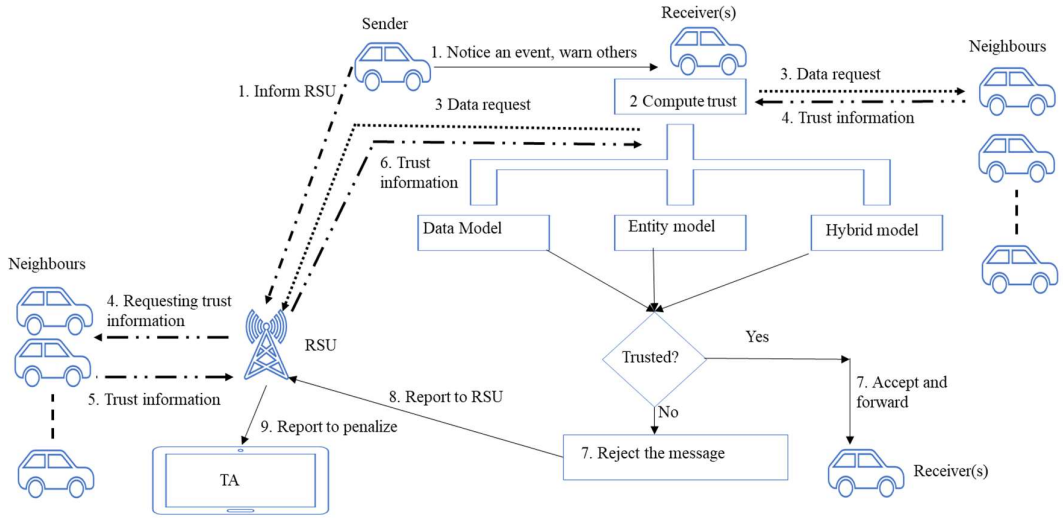


Fig. 2-4: A Generic Model for Trust Verification in Most Receiver End-Based Approaches

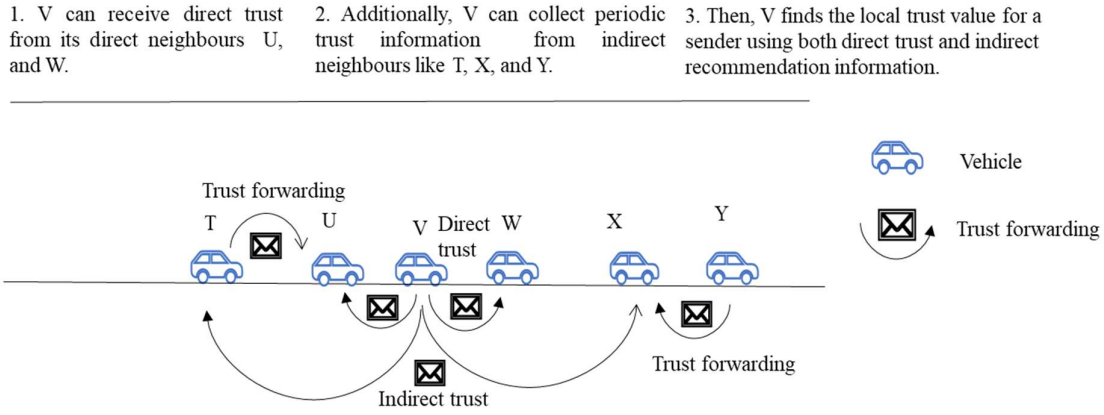


Fig. 2-5: Trust Calculation Using Direct and Indirect Trust

2.11 Summary

This chapter reviews existing trust management schemes. It discusses the issues, security requirements, threats, and approaches that can thwart attacks. Trust management is required to thwart attacks from authorized users. A significant amount of research has been proposed for trust management. However, existing methods exhibit limitations which are critically reviewed. The chapter also illustrates general communication patterns and issues with existing approaches. In the next chapter, the proposed trust management framework is presented.

Chapter 3: Sender-Side Regulated Trust Management Framework for VANETs

3.1 Introduction

In recent years, researchers have shown interest in trust management besides cryptographic solutions for VANETs. This comes from the inability of security approaches to thwart malicious attacks from authorized vehicles. Besides, research to-date has not agreed upon an acceptable trust model. Also, it is clear from Chapter 2 that most existing trust models exhibit performance problems, incurring additional delay while managing the trust of vehicles.

The most obvious problem found in existing trust models is verifying the trust score of a sender vehicle and its messages after receiving them along with uncontrolled message announcements. Trust verification often involves communication with neighbouring vehicles, RSUs, and the cloud which produces a huge communication overhead. Response time is also high as receiver vehicles wait for the updated information and/or the trust evaluation. The next problem is the inefficient utilization of bandwidth as vehicles with poor trust scores can still transmit messages. Considering these inefficiencies, this chapter proposes a new methodology for trust management which reduces a receiver's trust computational complexity after receiving messages as well as placing control on a sender's message announcements. Hence malicious message dissemination can be reduced so that the launching of attacks can be reduced. The adversary models considered with the proposed framework are the untrue attack, inconsistent attack, and cooperation attack. This approach thwarts these attacks by first detecting them, and then issuing punishment to demotivate the driver / vehicle from further malicious announcements. Additionally, this framework supports access blocking of an individual driver which is equivalent to the blacklisting in other approaches.

Ideally, a trust model promoting security should produce little or no extra burden in terms of computational and communication cost as vehicles move quickly. In a VANET, vehicles typically meet each other randomly and fleetingly. Thus, there is little time for decision-making based on trust. With receiver-side trust models, vehicles with a poor trust score can still send messages although these will typically be ignored by receivers once their trust level is discovered. However, this takes time, so vehicles may unnecessarily experience events such as traffic jams. To this end, our research proposes a novel sender-side trust management framework that reduces the amount of trust information passed over a VANET and blocks untrusted transmission attempts. In the next section, the trust management framework is introduced. The following is a list of features of the proposed trust management model:

- This model controls the broadcasting from the sending side based on the trust score of vehicles so a receiver vehicle can believe in a message instantly and does not need to take any further action.
- By regulating the ability to broadcast, malicious vehicles, once identified, are unable to broadcast false / untrue messages. Though, blacklisting is present in most existing approaches it requires the trust score to reach zero. Therefore, a malicious vehicle can create many hazardous problems in before being blacklisted.
- Response times are reduced as trust does not need to be verified on a per message basis. This also reduces the communication overhead.
- It uses the RSU for trust verification, when needed, rather than approaches which gather indirect trust from surrounding neighbours, and may receive false recommendations from malicious vehicles.

3.2 System Assumptions

The framework is based on a sender-side Tamper-Proof Device (TPD) fitted to regular vehicles to prevent unauthorized access and regulate transmissions based on the level of trust. Regular vehicles can also have a wireless card reader installed to support driver authentication and to store each driver's trust score. With this card, a driver could drive multiple vehicles and receive suitable service access by touching his/her ID card on the card reader and be authenticated with the TPD. The security of the TPD is beyond the scope of the proposed framework as it relates to physical layer protection. Also, we do not consider other security aspects with this trust framework as we believe that existing security techniques can address authentication, privacy, and integrity. A security approach that supports these functionalities can be incorporated with the framework to confirm the authenticity of the driver and/or messages with other entities, secure the privacy of the driver. HMAC can be used for achieving integrity (Tangade et al [7]). For example, when a driver registers with the TA, the driver can obtain private and public keys to encrypt and decrypt messages and can obtain a pseudo-identity associated with his driver ID for securing privacy with other drivers (Tangade et al [7]). RSUs, official vehicles, and the Trust Authority (TA) are also considered fully trustworthy. Both the TA and TPDs are governed and owned by the Road Transport Authority (RTA). The resilience of the TA infrastructure is beyond the focus of this work. We assume a driver has a built-in dashboard with designated touch buttons to display the classes of message available given his/her current trust score and to generate specific emergency events for other vehicles. Different classes of message have an associated trust threshold to permit their broadcast. Since announcements are regulated by the sender's TPD, receivers can believe messages and the sender's trust instantly. Furthermore, a TPD can access GPS data to determine the location of the vehicle.

3.3 Registration, Access-Blocking, and Redemption

Drivers may register themselves with the TA directly using an online form with the vehicle plate number as vehicle ID and driver's license number as a driver ID. Since this is an external process, it is outside the scope of the framework. Alternatively, if the system chooses to send a registration message from the driver when they start initially; then the RSU forwards the registration and confirmation of registration to and from the TA.

RSUs send the decisions of disputed events to the TA to store in a driver profile database which keeps driver and vehicle information, event information, and the reward / punishment. When the TA receives a decision on a disputed event, it searches the driver profile database. If three malicious events have been reported within a limited timeframe, the TA sends a blocking confirmation message to the RSUs in the driver's vicinity. When the vehicle receives a blocking confirmation message, the TPD blocks the access of the driver and acknowledges the blocking confirmation message to the TA via an RSU. The blocked driver can only send/receive beacons into the VANET. Additionally, a blocking message can be generated from the vehicle's TPD when the driver's trust score crosses the lowest acceptable trust limit. This message is forwarded by an RSU to the TA and the same mechanism is followed. By default, regular drivers obtain access to traffic data in the VANET with a trust score between 0.06 and 0.9. Whenever the trust score becomes less than or equal to 0.05, an external mechanism requires the driver to communicate with the TA to obtain redemption from blocking. We assume this is within the jurisdiction of the RTA and may involve issuing a monetary penalty or other sanctions.

3.4 Reward and Punishment Policy

Drivers improve their trust score by valid message announcements, forwarding, beaconing, untrue attack reporting and clarifying events to an RSU. These rewards are awarded by the TPD of the vehicle. Rewards from announcements are withheld for a period but punishments for untrue announcements are executed immediately. TPD reward for an announcement varies based on the travel distance from the event location to the announcement location and the elapsed time after first noticing the traffic event on a road. For example, a TPD assigns an announcement reward of between 0.01 to 0.08 based on the distance and time metrics and this procedure is detailed in the case [PosDiff | D] of Algorithm 3-2. Rewards for beaconing, forwarding, and clarifying to an RSU are also given instantly. A beaconing reward is given for each emitted beacon (0.0001 when trust is less than 0.25, otherwise 0.001) and a relaying reward is also granted (0.002). These rewards are very low as vehicles periodically send beacon messages and relaying is common. The reward for untrue attack reporting is fixed (0.08) and given when the RSU dispute ruling arrives at the TPD. Also, a driver builds his / her trust from RSU rewards whenever he / she "wins a dispute" over another vehicle. A driver receives a reward of 0.08 when he/she wins a dispute, and this is set to the maximum announcement reward from the TPD. Also, the TPD can

punish drivers when announcements are delayed, or a vehicle travels more than a specific distance before reporting a serious event. In some cases, a driver may be involved with an activity, for example, his/her passenger safety while he / she is passing the event location. Considering this situation, the framework does not punish drivers who ignore traffic events in the network as we cannot differentiate when a driver ignores announcing a traffic event or when he / she might be busy after he observes a situation. Drivers receive RSU punishment whenever they "lose a dispute" with other vehicles. RSU punishment is set higher (0.1) than the TPD or RSU reward; this is to uphold the principle "trust building is harder than to lose". This is also implemented in the trust model Zhang et al [123].

If regular vehicles spread untrue messages multiple times, they receive incremental punishments from the RSU (for example, 0.1, 0.3, 0.5 for three consecutive untrue announcements) using the Incremental Punishment Policy (IPP). These trust thresholds are selected to ensure the access-blocking of a driver when he / she announces three untrue messages consecutively. However, these punishments cannot ensure access blocking when drivers switch their behaviour by announcing both true and untrue messages as their trust will increment and decrement. Hence, a policy of access-blocking when a specific number of untrue announcements are made within a certain period can ensure even drivers who switch behaviour can still be blocked. One instance of the policy might be: if an untrue announcement happens thrice for severe cases, for example, an accident, then the network access of the driver is blocked, and he / she can only generate beacons until a redemption procedure is undertaken. The beaconing reward is only given when a driver is not access-blocked and the trust of a driver is less than 0.5. Similarly, a relaying reward is not given when trust score is greater than or equal to 0.5.

3.5 Framework Components

Both regular vehicles and official vehicles are present within the framework. The framework is extensible so that more vehicle classes could be added. The approach supports the same vehicle accommodating multiple drivers via individual driver trust management. The actions and responsibilities of each vehicle type are limited to their role. Every vehicle is pre-equipped with a built-in On-Board Unit (OBU), comprising a GPS unit for location access, a transceiver to communicate with other entities, and a TPD that manages the trust and regulates transmissions. We define the following actors based on their roles:

- Senders: are drivers that can originate both true and untrue announcements relating to an incident, such as an accident, subject to their trust score. For a true announcement, the trust manager within the TPD rewards the driver if the claim is not disputed within a given time.
- Reporters: are drivers that refute an announcement of a sender and receive a reward or punishment if the challenge is confirmed or dismissed, respectively. If they do decide to make a report, they may do so either truthfully or falsely. Failure to make a report carries no penalty.

- Receivers: are drivers that receive messages from any entity and relay them automatically provided the hop limit is not reached and their trust is sufficient.
- Clarifiers: If a dispute is detected at an RSU, the RSU transmits a query message seeking clarification concerning a disputed incident. Vehicles that receive this message can choose to answer the query, i.e., to respond to the RSU, confirming or denying that the incident has taken place, or ignore it. If they respond, they are considered clarifiers.
- Road-Side Units: are automated units that receive information from senders, reporters, and clarifiers, either directly or via intermediate vehicles that rebroadcast the received messages. If information from multiple senders, reporters, or a combination of these conflicts, then the RSU will rule on the dispute. RSUs act as an intermediary between the vehicles and the TA.
- Trust Authority: is the ultimate authority in this framework which validates registration and access-blocking of drivers. The TA blocks access of a driver whenever it receives an access-blocking message initiated from a TPD or if it finds Three Malicious Events (3ME) for the same driver within a configurable timeframe. The TA then replies with a blocking confirmation to the RSUs in the vicinity of the last disputed event to reach the vehicle's TPD. Incidents reported by RSUs are saved by the TA in an incident database including the location, timestamp, and incident information. The TA also maintains a driver profile database containing the reward/punishment history of drivers.
- Official Vehicles: This framework considers police, ambulance, and fire service vehicles as official vehicles. Their primary task is to respond to emergency issues on roads by cooperating with RSUs. They are always trusted.

3.6 Trust Evaluation Mechanism

3.6.1 Trust Score of Regular Vehicles

The framework only computes the trust score of regular vehicles. Trust is the degree of belief that can be represented by values between 0 and 1. Here, a 0 value of trust means absolute distrust and a value of 1 is complete trust. In this framework, different trust thresholds are selected in the following way: the framework divides the whole trust range (0 to 1) linearly. The lower trust range (0 to 0.49) belongs to untrusted and partially trusted vehicles whereas, the upper trust range (0.5 to 1) is reserved for trusted and highly trusted vehicles. The network access for all traffic event reporting requires a trust level of 0.5 upwards. Various trust thresholds are selected after reviewing existing trust models though slightly changed values from these models are adopted. The reasons behind these adjustments are explained next.

Mühlbauer et al [14] assign reputation scores (0 to 10) to vehicles. It initializes a reputation value of 5 while conducting analysis. This is one example which uses a middle value as the initial reputation. Other trust frameworks (Rostamzadeh et al [20], Wagan et al [46], and Kerrache et al [125]) also use 0.5 as their initial trust score with the lowest and the highest trust being 0 and 1, respectively. Gazdar et al [12] use 0.5 as an initial trust score to avoid the cold start problem. These models do not allocate

beacon rewards. In contrast, the proposed framework operates with a normal trust range of 0.06 to 0.9, allocates beaconing rewards, and reserves the trust value of 1 for official vehicles. This framework selects 0.45 as the initial trust score which also avoids the cold start problem as it is very close to 0.5 (with a trust of 0.5, a vehicle can announce any event). The initial trust of 0.45 is chosen such that vehicles join with a limited announcement capability preventing them from sending severe event announcements without knowing much about the consequence of sending untrue messages. Within this period, they learn about the system, relay other's events, and send beacons to get rewards until they earn a trust score of 0.5. The trust score of 0.5 is particularly important in this framework as with this trust score, all the supported events are available on the driver's screen.

With the highest trust=0.9, it enables many possible trust values between 0.5 to 0.9 to be used with the trusted regular vehicles. If the framework limits the highest trust to 0.6, it is easy to reach this trust level by malicious vehicles which is not desirable as drivers with bad intentions may reach this trust first to launch malicious attacks after earning trust. After that they may change their behaviour and the framework will treat them as highly trusted. This can be resolved by setting a higher trust limit. A trust range with many possible values offers many values to assign to the vehicles which helps to differentiate their trustworthiness more easily. Also, it contributes to obtaining different weighted feedback when an RSU validates an event. Kerrache et al [125] assign a trust value of 1 to police and 0.8 to ambulances. Our framework assigns trust score of 1 to official vehicles (including police, ambulance, and fire service vehicles) as they are part of the authority. This framework keeps a significant trust score gap between the highest trust score of regular vehicles and official vehicles.

Alternatively, the lowest trust is the absolute distrust which is used by most trust frameworks (Mühlbauer et al [14], Rostamzadeh et al [20], and Yao et al [144]) but they do not allocate beaconing reward. Our framework does not utilize some trust levels i.e., 0, 0.01, 0.02, 0.03, and 0.04. As this trust level is very close to absolute distrust which is 0. The framework blocks access of a driver whenever trust reduces to 0.05. Selecting 0.05 rather than 0 allows drivers to be access-blocked slightly earlier. As a result, the framework disables activities from malicious drivers earlier. After access blocking vehicles / drivers need to undergo an external redemption procedure, or they can be charged a monetary penalty which is under the jurisdiction of the RTA.

The trust score for accessing all events is set to 0.5 which is the middle value of the whole trust range. If this limit is set to 0.6 when the number of events is low, it is expected that only limited vehicles may achieve this score to become trustworthy in the network. Thus, accessing all events will be delayed if vehicles start with an initial trust score of 0.45. In contrast, if the trust score for accessing all events is set to 0.4, then a low trusted driver will gain full access prematurely. Trusted vehicles / drivers can be assigned many possible values from 0.5 to 0.9 rather than having a trust range like (0.6 or 0.7 to 0.9). This way the system can differentiate their trust with more granularity. Furthermore, relaying of traffic

event messages is disallowed when trust is less than 0.25. Vehicles with trust range of 0.26 to 0.49 can only announce a restricted set of events where events are grouped into tiers. Thus, our framework provides a restricted announcement capability when trust becomes lower than 0.5 and revokes announcement of all traffic events when trust is less than 0.25. Furthermore, it revokes the relaying of events when trust score is less than 0.25. Also, beaconing, and relaying reward is not given when trust score is higher than 0.49.

Abassi et al [127] blacklist vehicles with trust value of 0 and do not allocate periodic beaconing rewards. Similarly, when there is no credit left approaches like Rai et al [151] and Liu et al [152] revoke vehicle participation from the network. Also, Mühlbauer et al [14] allow redemption when the reputation of vehicles reach 0 though this scheme considers certificate revocation outside the scope of the work. A dynamic threshold value is used in Dahiya et al [10] to blacklist vehicles. In contrast, the framework does the access-blocking at 0.05 (which is one-tenth of 0.5 where the vehicle achieves access to all events) in the presence of beaconing rewards.

Our framework selects 0.45 as the starting point. Setting an initial trust score of 0.45 avoids the cold start problem and encourages participation from vehicles. From trust value of 0.5 to the highest value of trust score the framework supports, the driver / vehicle remains trusted, and all the services are available. Regular vehicles can achieve a maximum trust score of 0.9. Conversely, when their trust score is less than 0.25, vehicles / drivers cannot relay messages from others. If their trust score is lower to 0.05, then their network access for all traffic events is blocked other than for beacon transmissions. The trust thresholds are shown in Table 3-1 and the trust T of a regular vehicle/driver i is expressed by Eqn (3-1).

$$T_i = \{t \mid t \in R \mid 0.05 < t \leq 0.9\} \tag{3-1}$$

Table 3-1: Trust Thresholds

Trust Thresholds	Explanation
T=0.45	Initial trust score of regular vehicles
T>=0.5	Enables the possibility to announce all supported events in a VANET
T<0.25	Relaying is not permitted
T<=0.05	The TPD sends an access-blocking request to the vehicle application layer to broadcast it. The TA first confirms the access-blocking and then the TPD of the vehicle implements the access-blocking

The following equations define the level of access control with this framework. Eqn (3-2) limits the trust score of a driver to the range of 0.05 to 0.9 if it attempts to go beyond the upper and lower thresholds after performing trust adjustments. Eqn (3-3) expresses access-blocking of a driver / vehicle. Eqn (3-4) expresses the condition to achieve the message relaying capability. The Eqn (3-5) specifies the announcement ability of regular vehicles.

$$T_i = \begin{Bmatrix} 0.05, & T_i < 0.05 \\ 0.9, & T_i > 0.9 \\ T_i, & 0.05 < T_i \leq 0.9 \end{Bmatrix} \quad (3\text{-}2)$$

$$T_i = \{Blacklist \quad T \leq 0.05\} \quad (3\text{-}3)$$

$$Message\ Relaying - ability = \begin{Bmatrix} False & 0.05 \leq T < 0.25 \\ True & T \geq 0.25 \end{Bmatrix} \quad (3\text{-}4)$$

$$Message\ \ Generation - ability = \begin{Bmatrix} Limited & 0.05 \leq T < 0.5 \\ All & T \geq 0.5 \end{Bmatrix} \quad (3\text{-}5)$$

Within this framework, regular vehicles are classified as *blocked* (T = 0.05), *not trusted* (0.05 < T < 0.25), *lowly* trusted (0.25 ≤ T < 0.5), *trusted* (0.5 ≤ T < 0.8), and *highly trusted* (0.8<T≤0.9). Since the framework accommodates official vehicles, they are assigned a trust score in excess of regular vehicles (i.e. T = 1.0) as a regular vehicle should never be trusted more than an official vehicle.

3.6.2 Trust-Based Access Control for Message Announcements

We envision a driver's dashboard as consisting of a set of buttons for supported actions in the framework as shown in Fig. 3-1. Appropriate buttons can be pressed relevant to a specific type of road incident from the screen. The buttons are enabled based on the driver's trust score. There are three main classes of message in the hierarchy and each of these classes has an associated trust score which is checked whilst attempting an announcement. The lowest class consists of beacons and "wave" service announcements represented by red-coloured messages, though a blocked driver cannot use the "wave" service facility. The next class of messages consists of announcements of poor road conditions, debris, road defects, and so forth. This class of messages is the black-coloured messages in Fig. 3-1. These can only be broadcasted by drivers with a trust score greater than or equal to 0.25 and less than 0.49. The highest class of messages consists of announcements for accidents, traffic jams, road closures, etc., as well as untrue attack reporting messages. This class of messages is the blue-coloured messages in Fig. 3-1. To announce a message from this class, a driver must have a trust score of at least 0.5. This Fig. also demonstrates the communication between the driver application and the TPD to transfer trust score, reward, punishment, and metrics for reward assessment from the announcements. Additionally, it indicates other elements of an OBU including a GPS unit and a wireless transceiver.

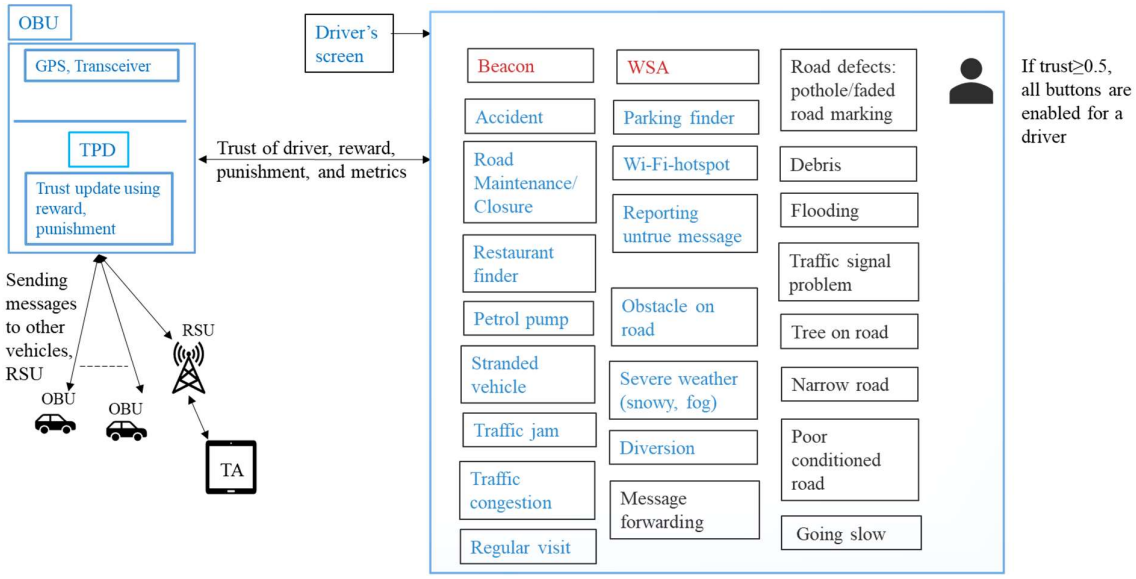


Fig. 3-1: Classes of Message on the Dashboard

3.6.3 Traffic Event Management of Regular Vehicles

A driver may or may not announce a traffic event when he / she first observes it. When a driver ignores a traffic event, he / she does not receive a punishment as he/she might be involved in dealing with his/passenger issues. We believe another driver will announce the event when he / she notices no other driver announces it before. Normally, a driver announces an event when he / she observes it for the first time. It also receives traffic events from other vehicles and relays the messages to its neighbours. Message dissemination is achieved through relaying of a message up to a configurable hop limit. The framework employs this flooding mechanism to reach possible affected vehicles in the vicinity of an event. With an announcement, many neighbouring vehicles may avoid a traffic jam for example. These messages also arrive at RSUs and official vehicles. These entities respond differently based on the severity of traffic events. This subsection covers only the announcement, retransmission, relaying, feedback, and reporting activities of regular vehicles which is shown in Algorithm 3-1. Notations and symbols for Algorithm 3-1 are provided in Table 3-2.

Table 3-2: List of Notations

Notation	Meaning
RSU_r	r^{th} RSU
$T_s(D_s(V_s)$	$Trust_s$ of driver s of vehicle s
V_{rep}, V_{rec}, & $V_{trust\text{-}cla}$	Reporter, receiver, and trusted clarifier vehicle
$timer_{reward\text{-}withhold}$	When to process reward/punishment
evt_e and untrue(evt_e)	Traffic event and reporting the evt_e
RSU_{clarif_query}	Clarification query from RSU
T_{dis}, and T_{int}	Time threshold to send feedback and to report evt_e

TTL, and M_{cls}	Time-To-Live, class of messages
$ATT(M_{cls})$	Associated trust threshold of M_{cls}
Rew_r/Pun_r and Rew_{tpd}/Pun_{tpd}	Reward/punishment from a RSU_r, and TPD
HL and RT_L	Hop limit and retransmission limit
$Reward_f$, $Reward_{clar}$, and $Reward_{unt\text{-}atck}$	Reward for forwarding, clarification, and reporting
$LowTrust_{msg}$	Forwarding is not possible with low trust
$timer_{bilst}$	To check blocking condition meet
driver_List	Registered driver list
$Trust_s$, $Trust_d$	Saved and initial trust
Complaint_List	List of reported announcements
longDelayed	Driver delayed than the upper limit
Msg_{block} & $Msg_{block\text{-}conf}$	Blocking and blocking confirmation message
PosDiff	Distance between the event and announcement location

Algorithm 3-1: Regular Vehicle Traffic Event Management

Input: Driver ID, Vehicle ID, events, trust of drivers, hop and retransmit limit
Output: controlled broadcasting, relaying, reporting, and sending feedback

1. **case *eventType*** of
2. ***witnessed-event*:**// to warn others.
3. **if** ($T_s(D_s(V_s)) \geq ATTL(evt_e)$)
4. $D_s(V_s)$ prepares and broadcasts the evt_e
5. Send metrics to TPD to find Rew_{tpd}/Pun_{tpd}
6. **end if**
7. ***reported-event*:**// to report the received event.
8. **if (**V_s decides evt_e=false) and (V_s visits event location within T_{int}) and ($T_s(D_s(V_s)) \geq 0.5$)
9. Send *untrue*(evt_e) towards RSU
10. Notify TPD to add $T_s = T_s + Reward_{unt\text{-}atck}$
11. **end if**
12. ***relayed-event*:**// to relay event up to hop limit.
13. **if** (V_s gets an evt_e or an *untrue*(evt_e) from a V_{rep} first time)
14. **if** (V_s sends evt_e or *untrue* (evt_e))
15. Return
16. **end if**
17. **if (**$T_s \in (T_s > 0.05$ and $T_s <= 0.25)$
18. Send a $LowTrust_{msg}$
19. **else**
20. **if** TTL(evt_e or *untrue*(evt_e) $\geq$ HL)
21. Stop resending evt_e or *untrue*(evt_e)
22. **else**
23. Resend evt_e or *untrue*(evt_e) up to HL
24. Notify TPD to add $T_s = T_s + Reward_f$
25. **end if**
26. **end if**
27. **end if**
28. ***retransmit-event:*** // to repeat the broadcasting
29. **if** (no_of_time $\leq RT_L$)
30. Resend evt_e

31. **end if**
32. ***feedback-event:*** // to send feedback.
33. **if** (V_s receives a RSU query about evt_e)
34. **if** (V_s is a V_{rep} or is the sender of evt_e)
35. Return
36. **end if**
37. **if** TTL(RSU_{clarif_query}) < HL
38. Resend RSU_{clarif_query} message
39. **end if**
40. **if** (V_s visits the event location within T_{dis})
41. Send feedback
42. Notify TPD to add $T_s = T_s + reward_{clar}$
43. **end if**
44. **end if**
45. **end case**

3.6.4 TPD Reward and Punishment Assessment

A potential reward is initially assessed at a TPD and then withheld for a period before adding to the current trust. The withhold time is long enough to receive a complaint from another driver if there is one nearby. This duration should not be very long as the increases the decision latency, but it should be long enough for the verification to be accurate. If this period is too short (for example, 30 seconds) then a sender might be given a reward before complaints are resolved. To avoid this situation, a two-minute reward withhold timer is applied. It is expected that a vehicle will visit the event location within this two-minute or, otherwise, the announced event will not intensify the severity of a situation in the absence of vehicles. During this time, a TPD waits until a timer expires to check any complaint has been raised by any reporter. If there is no complaint, then the TPD adds the reward to the current trust of the driver. Rewards are calculated based on message accuracy (no complaint), location difference, and delay / responsiveness. Thus, the framework promotes emergency event announcements at the earliest possible opportunity. The distance a vehicle moves between the event location and the vehicle's current position is passed to the TPD to evaluate the location difference. Delay is calculated as the difference between the announcement and the observation time (driver can initiate a timer to record the observation time) on the road. The TPD uses this information to assess the reward/punishment for the announcement. Also, the framework suggests a vehicle should not travel more than 500 meters or the next traffic signal in order to earn a higher reward from the announcement. The trust T_i is updated inside the TPD using Eqn (3-6).

$$T_i = T_{i-1} + R_i \text{ “or” } P_i \tag{3-6}$$

Here, T_i is the revised trust score of a vehicle after adding a reward/punishment to its current trust T_{i-1}. R_i *“or”* P_i is the estimated reward or punishment for the i^{th} message announcement. The set of rules used by the TPD for deciding the appropriate reward/punishment magnitude for a given announcement is written in the “metrics” event handling in Algorithm 3-2. If the message is accurate and the driver travels less than 1200 meters from the incident or sends a notification within 120 seconds

then a reward is given. This reward is then withheld for a period and the reporting status of the announced message is checked before the trust update. If the message is accurate but the driver travels more than 1200 meters or notifies after 120s, then no reward or a nominal punishment is issued. If the assessment is punitive, then it is deducted immediately from the current trust. However, if the message is inaccurate (i.e., a complaint received during the reward withhold period), then the driver receives no reward for the announcement from the TPD and defers the reward/punishment decision to the RSU. As a VANET is a time-critical system, vehicles should disseminate information promptly. Thus, the delay and distance travelled are considered in the reward calculation besides the message accuracy. In Algorithm 3-2, the TPD trust update mechanism and access-blocking management are shown for regular vehicles. Notations and symbols for Algorithm 3-2 are taken from Table 3-2.

Algorithm 3-2: Trust and Blocking Management at the TPD

Input: Announced evt_e, reporting status of evt_e, $Msg_{block\text{-}conf}$, PosDiff, delay, Rew_r/Pun_r,
Output: Trust update and access blocking

1. **case eventType** of
2. ***periodic-blocking-checker***://
3. **if** $D_s(V_s)$ is unblocked) and ($timer_{blist}$ expires)
   and ($T_s(D_s(V_s))$ <=0.05))
4. TPD_s issue a Msg_{block} to reach TA.
5. **end if**
6. **if** $Msg_{block\text{-}conf}$ comes from TA for $D_s(V_s)$
7. Disable the network access for D_s
8. **end if**
9. ***RSU reward/punishment:*** // Add with trust.
10. **if** (Rew_r/Pun_r from an RSU_r for the $D(_s(V_s))$)
11. $(T_s(V_s(D_s))_= (T_s(V_s(D_s))+Rew_r/Pun_r$
12. **end if**
13. ***metrics*:** // Applies reward withholding.
14. **if** (evt_e =false by receiving a complaint)
15. $reward_{evt\text{-}e}$ =0
16. **else**
17. **case** [PosDiff | D]
18. 0<PosDiff<300m | 0<D<15s:
19. $reward_{tpd}$ =0.08.
20. 301<PosDiff<500m | 16<D≤30s:
21. $reward_{tpd}$ =0.06.
22. 501<PosDiff <800m | 31<D≤60s:
23. $reward_{tpd}$ =0.05.
24. 801<PosDiff<1200m | 61<D≤120s:
25. $reward_{tpd}$ =0.01.
26. 1201<PosDiff<1500m | 121<D≤150s:
27. $reward_{tpd}$ =-0.01.
28. longDelayed=true
29. PosDiff >1500m | D>150s:
30. $reward_{tpd}$ =-0.05.
31. longDelayed=true
32. **end case**
33. **if** longDelayed=true
34. call reward/punishment process immediately
35. **else**
36. start $timer_{reward\text{-}withhold}$ to process reward
37. **end if**
38. **end if**

39. ***TPD reward/punishment:***//add with trust
40. **if** (broadcasted msg_id ∈ Complaint_List)
41. Update $T_s(D_s(V_s) = T_s(D_s(V_s)$
42. Return
43. **else**
44. Update $T_s(D_s(V_s) = T_s(D_s(V_s)+reward_{tpd}$
45. **if** $(T_s(D_s(V_s))>0.9$
46. Update $T_s(D_s(V_s)=0.9$
47. **end if**
48. **if** $(T_s(D_s(V_s))\leq 0.05)$
49. Update $T_s(D_s(V_s)=0.05$
50. Start $timer_{blist}$
51. **end if**
52. **end if**
53. ***complaint-on-broadcasted-event*:**// save report.
54. **if** the broadcasted msg_id has a complaint
55. Save the complaint into the Complaint_List
56. **end if**
57. ***driver-change-event*:**// to change driver
58. Extract the driver_name D_s
59. **if** (D_s exists in the driver_List)
60. Use D_s as the current driver and use the $Trust_s$
61. **else**
62. Add D_s to the used_driver_List and $T_s=Trust_d$
63. **end if**
64. **end case**

3.7 Functional Diagram of the Proposed Framework

Assume a driver sees an incident and wishes to announce it as shown in Fig. 3-2. The framework first checks the trust score from the TPD and determines if the action is eligible with the driver's current trust. If this test is passed, then the driver announces the incident. Other "receivers" forward it up to the configurable hop limit. As this is an original announcement, the driver is classified as a "sender". If this announcement reaches subsequent drivers who visit the same location later, they can notice whether the said event has occurred or not. However, if any driver believes the announcement to be untrue, that driver can send a complaint to the RSU. When the RSU receives this complaint, the RSU requests "trusted clarifiers" to respond, confirming or denying the claim.

The RSU collects feedback from these trusted clarifiers who have recently visited the event location. After this, the RSU rules on the validity of the event and penalizes or rewards the respective vehicles. An RSU always informs the TA of the outcome of a dispute, which could be a driver being malicious. It is then up to the TA to check prior behaviour for three malicious activities from a specific driver over a configurable time and potentially block this driver. The TA sends a blocking confirmation message to RSUs in the vicinity from where the TA receives the last dispute decision. These RSUs broadcast it to the concerned vehicle which receives and acknowledges the instruction. Alternatively, if a driver's trust reduces to 0.05, the TPD generates an access-blocking request message which a nearby RSU relays to the TA. Then the TA blocks this driver and informs the respective TPD via the RSUs in the vicinity of the vehicle.

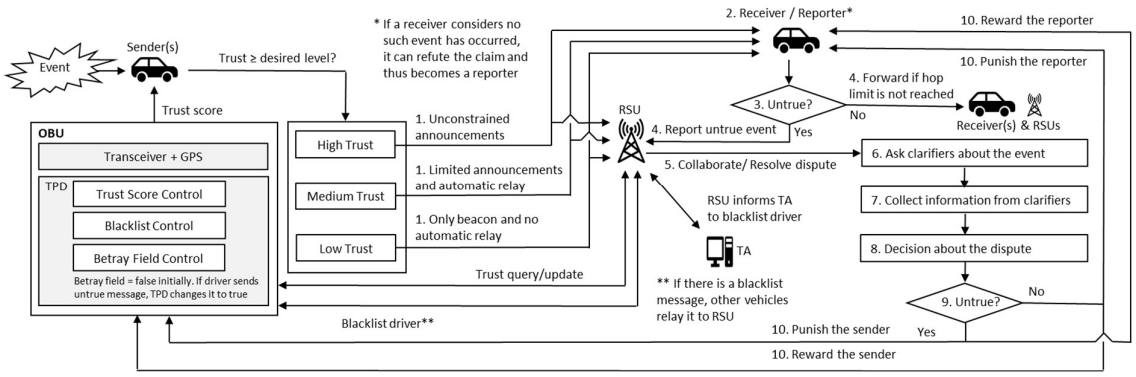


Fig. 3-2: Functional Diagram of the Proposed Trust Management Framework

3.8 RSU Traffic Event Management / Functionality

RSUs always listen to traffic events and share them as necessary based on their severity. RSUs receive regular beacons from vehicles. In response, RSUs send beacons periodically to notify of their roadside existence so that vehicles can request services. When RSUs receive an emergency traffic event message from a sender vehicle, they rebroadcast the same towards the neighbouring vehicles so that oncoming vehicles whose route includes the problematic road may avoid it. RSUs also share certain events with nearby RSUs so that vehicles in a greater region may avoid the problematic road, if appropriate. For some events, RSUs will continue to make periodic announcements until they receive notification from an official vehicle to confirm the event is resolved. When a traffic event occurs and if the RSU receives an AttendingBY-V_{off} message from an official vehicle, it confirms the event to an RSU. The RSU continues to announce the traffic event periodically until the reception of a traffic event sorted message from the official vehicle. When the event-sorted message arrives, the RSU stops rebroadcasting the original traffic event. Rather it starts broadcasting the sorted traffic event up to a retransmission limit as well as relaying this message to nearby RSUs based on the severity of the original traffic event. RSUs rebroadcast and relay traffic incidents to the TA from sender vehicles besides storing traffic incident information until it is resolved. Each local service point, for example, petrol pumps, and car parks is registered in advance with the nearby RSU. Whenever any vehicle sends any query seeking information regarding any service, the local RSU sends a reply to the service query containing the information of queried service or it says it has no information if it does not know.

An RSU assigns a fixed amount of reward and sets the punishment for disputed announcements using an Incremental Punishment Policy (IPP). An RSU forwards the decision of a disputed event to the TA. Then the TA checks the malicious event count for relevant drivers. If the 3ME condition holds, then the TA sends a blocking confirmation message to the RSUs in the vicinity of the last disputed event. After that, relevant RSUs broadcast this message to the vehicle. Besides these functionalities, an RSU also resolves disputes when a reporter reports an untrue attack which is illustrated in Section 3.9.

These functionalities are described as the generic traffic event handler in Algorithm 3-3. Notations and symbols for this algorithm are listed in Table 3-2 and Table 3-3.

Table 3-3: List of Notations

Notation	Meaning
evt_e-sorted	Sorted event evt_e
untrueHandledList	List of disputed cases which RSU_r has decision
send-a-reply(evt_e)	Asks V_{cla}s to send feedback on evt_e
$Loc_{req}(service_i)$	Location of i^{th} service query
$Loc_{rep}(service_i)$	Location of i^{th} service reply
$RSU_{r\{known\ service\}}$	Registered services at RSU_r
$timer_{sc}$	Timer to collect feedbacks by an RSU
$reply_{off}$	Reply from V_{off}
IPP	Incremental punishment policy
3ME	Three malicious events
untrue_id	Untrue_attack message id
AttendingBY-V_{off}(evt_e)	Attending evt_e by a V_{off}
rewV, punV	Rewarded and punished vehicle

Algorithm 3-3: RSU General Traffic Event Management

Input: traffic event, service query

Output: send traffic updates, manages road traffic event, communicates TA and other RSUs, as necessary

1. **while** running
2. **case eventType** of
3. ***traffic-event:*** // deals with the received event
4. **if** (RSU_r gets unique evt_e from V_s or RSU_s)
5. Store evt_e and forward to TA
6. **if** TT(evt_e)< HL
7. Rebroadcast evt_e
8. **end if**
9. Send *avoid-road* periodically until an evt_e-sorted comes from a V_{off}
10. **if** evt_e comes from a V_s
11. forwardMsgtoRSU$_s$(evt_e) based on severity of event
12. **end if**
13. **end if**
14. **if** RSU_r gets an Attending-V_{off} (evt_e) from V_{off}
15. RSU_r expects V_{off} sends an evt_e-sorted soon
16. **end if**
17. **if** RSU_r gets an evt_e-*sorted* from a V_{off}/RSU_s
18. Stop rebroadcasting of evt_e
19. **if** evt_e-*sorted* from a V_{off}
20. forwardMsgtoRSU$_s$(evt_e-*sorted*) based on severity of event
21. **end if**
22. **if** ($no\_of\_time \leq RT_L$)
23. Retransmit evt_e-*sorted*
24. **end if**
25. **end if**
26. ***retransmit-event:*** // to retransmit same event
27. **if** ($no\_of\_time \leq$ HL)
28. Retransmit evt_e
29. **end if**

30. ***service-event:*** // replies to the service query
31. **if** (RSU_r gets a $Loc_{req}(service_i)$ and $service_i \in RSU_{r\{known\ service\}}$)
32. Send a $Loc_{rep}(service_i)$ with the information
33. **end if**
34. ***blocking:*** // deal with blocking
35. **if** (Msg_{block} comes from a V_s)
36. forwardMsgtoTA (Msg_{block})
37. **end if**
38. **if** $Msg_{block\text{-}conf\ (V\_ID+DRI\_ID)}$ arrives from TA
39. Send $Msg_{block\text{-}conf\ (V\_ID+DRI\_ID)}$ to vehicle when nearby or resend up to HL
40. **end if**
41. **end case**
42. **end while**

3.9 RSU Untrue Message Detection

If an RSU receives conflicting information from a sender and a reporter, it initiates a "collaboration" process to determine the validity of the disputed event. To this end, first, an RSU broadcasts a send-a-reply message to all trusted clarifiers in the vicinity including possible official vehicles and waits for a timer to expire when the feedback collection is finished, as depicted in Algorithm 3-4. Notations and symbols for this algorithm are listed in Table 3-2 and Table 3-3. Sender(s) and reporter(s) involved in the dispute are not permitted to participate in this clarification process. It is reasonable to consider that there are some trusted vehicles around the event. Also, there may be several malicious vehicles (Wei et al [8] and Yang et al [83]). The effect of malicious feedback will be nullified when the true feedback outweighs the malicious feedback when taking a decision. In this framework, feedback can only be generated by trusted clarifiers with trust scores greater than 0.5 and official vehicles. The possible feedback messages are 'YES' or 'NO'. Eligible vehicles that respond, known as clarifiers, reply 'YES' if they had visited the event location recently and confirmed the event or 'NO' if they had visited the event location and did not see the event. In some cases, drivers neither notice the event nor visit the event location in the recent past. These drivers will simply ignore the RSU query. Also, official vehicle feedback is treated as the decider for a dispute which bypasses the collaboration process since collected feedback from the trusted clarifiers is not used in forming a decision. When an RSU receives official vehicle feedback in Algorithm 3-4, it instantly invokes the reward-punishment generator as shown in Algorithm 3-5.

The RSU dispute resolution mechanism in Algorithm 3-4 uses the feedback to decide the truthfulness of a dispute. Here, the RSU performs a sum of product calculation of the feedback and the trust of the clarifiers to decide on the disputed event. Fig. 3-3 shows a driver's screen showing the opinion to generate, and an RSU collects feedback in a table. For example, suppose a vector of feedback is ('YES', 'YES, 'NO', 'NO', 'YES') which are represented programmatically as (1, 1, -1, -1, 1) and the clarifier's corresponding trust scores are: (0.5, 0.7, 0.65, 0.68, 0.9), then the RSU decides by using

Eqn (3-7). It should be noted that only trusted clarifiers can join the collaboration process. Generally, Eqn (3-7) can be expressed as in Eqn (3-8) for n feedbacks collected from the n trusted clarifiers, where F_i is the i^{th} feedback and T_i is the i^{th} clarifier's trust score.

$$Decision = [1*0.5] + [1*0.7] + [-1*0.65] + [-1*0.68] + [1*0.9] \quad (3\text{-}7)$$

$$Decision = \sum_{i=1}^{n} F_i * T_i \quad (3\text{-}8)$$

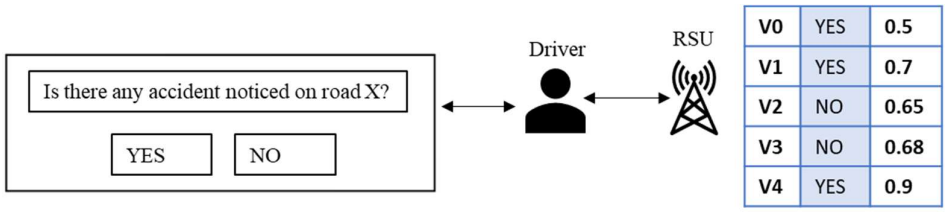


Fig. 3-3: Feedback Collection at an RSU and the Driver's Screen

If the outcome / decision is positive, then the RSU decides the sender has disseminated a true event and thus receives an RSU reward; the conflicting reporter(s) receive an RSU punishment. If the outcome is negative, the converse actions are followed. When the decision is reached, the RSU calls the reward-punishment generator, shown in Algorithm 3-5. During the punishment assessment, the IPP is adopted to influence the future good behaviour of drivers. However, for an unresolved issue when the RSU has no feedback data or Decision=0 in Eqn. (3-8), the RSU stores it in an unresolved dispute list and later may ask an official vehicle to inspect the event location physically and report its findings so that the RSU can take action on the dispute. The overall untrue message detection process is depicted in Fig. 3-4. It should be noted that if during the collaboration process, any official vehicle receives an RSU message, but they have not visited the disputed event location recently, then they reply with a far-from-event message. However, if the RSU receives a decisive message from an official vehicle, then it always decides on the event using this message and bypasses the collaboration mechanism.

Algorithm 3-4: RSU Untrue Attack Handler

Input: untrue attack, feedback message, trust, lists to save events
Output: initiate feedback collection for a timer, find rewarded/punished vehicle

```
1.  while running
2.    case eventType of
3.    untrue-attack: // deals with untrue attacks.
4.    if unique untrue(evt_e) from RSU_s/V_s
5.      Insert into untrueAddedList
6.      if untrue(evt_e) ∈ untrueHandledList
7.        Return
8.      else
9.        Insert into untrueHandledList
10.       if untrue(evt_e) from a V_s
11.         Broadcast a send-a-reply(evt_e)
12.         Start a timer_sc to collect feedbacks
13.       end if
14.     end if
15.   else
```

```
16.     RSU_r receives an untrue(evt_e) from a V_off
17.     Call rew-pun-generator(V_off, V_s)
18.   end if
19.   feedback: // collect all the feedbacks.
20.   while (timer_sc is not expired)
21.     if unique feedback f_u from V_cla is for RSU_r
22.       Insert in vector <f_0, f_1, … f_n>
23.       if f_u is from a V_off
24.         if f_u is the same as the V_s's event
25.           Call rew-pun-generator(V_s, V_rep)
26.         else
27.           Call rew-pun-generator (V_rep, V_s)
28.         end if
29.         Update rewardList and punishmentList
30.         forwardMsgtoRSU_s(decision_untrue)
31.         if count(3ME(V_s or V_rep)) ≥ 3
32.           Send a Msg_block(count(3ME(V_s or
              V_rep) ≥ 3)) to TA
33.         end if
34.       end if
35.     else
36.       The feedback is for different RSU_s
37.     end if
38.   end while
39.   decision-of-untrue: // to resolve dispute
40.   if timer_sc expires
41.     if the untrue(evt_e) has a decision
42.       Return
43.     else
44.       Sum=0
45.       case feedbackType of
46.         Positive:    F_i =1
47.         Negative:    F_i =-1
48.         Unsure:      F_i =0
49.       end case
50.       for each F_i from feedback vector <F_n ,T_n>
51.         Sum +=T_i*F_i
52.       end for
53.       if Sum>0
54.         V_s send true event, V_rep send false report
55.         Call rew-pun-generator(V_s, V_rep)
56.       else if (sum<0)
57.         V_s send false event, V_rep send true report
58.         Call rew-pun-generator (V_rep, V_s)
59.       else
60.         Undecided conflict
61.         Insert attack into unresolvedUntrueList
62.         Send an unresolvedUntrue(evt_e) to a V_off
63.       end if
64.       if (sum>0 or sum<0)
65.         Call forwardMsgtoRSU_s(untrue_ dec)
66.       end if
67.       Clear the vector<feedback> on untrue_id
68.     end if
69.   end if
70.   end case
71. end while
```

Algorithm 3-5: Reward-Punishment-Generator (rewV, punV)

Input: rewarded vehicle, punished vehicle
Output: send reward/punishment message and blocking message to TA, if required

1. **while** running
2. ***reward/punishment:*** //estimate reward or punishment for disputed event
3. Store reward(rewV) and punishment(punV) in rewardList and punishmentList
4. **if** (rewV!=V_{off})
5. Send the reward_msg(rewV)
6. **end if**
7. Send the punishment_msg (punV)
8. Call forwardMsgtoTA(untrue_dec)
10. **end while**

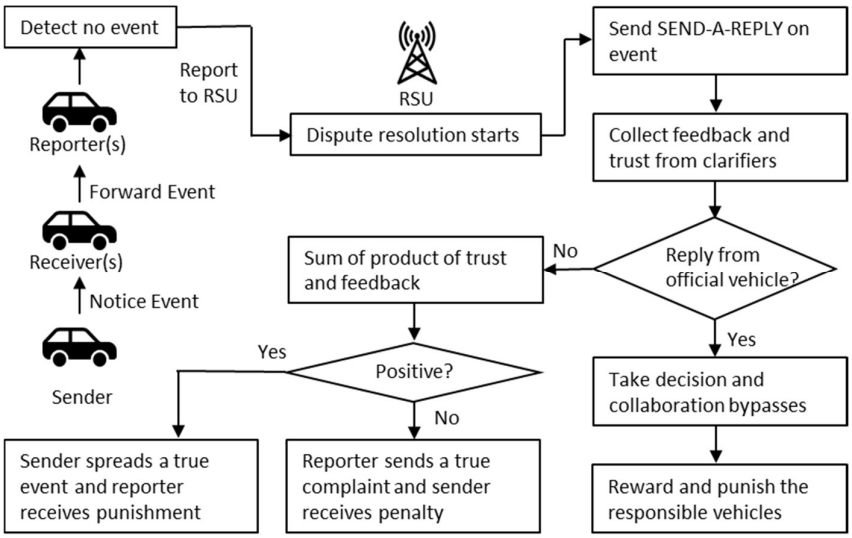


Fig. 3-4: RSU Steps for Untrue Message Detection

3.10 Pattern of Communication in the Proposed Framework

Fig. 3-5 depicts the type of messages various entities exchange with the proposed trust management framework. It is clear from Fig. 3-5 that all entities except the TA can exchange WSM (WAVE Short Message) and beacon messages. Only the RSUs can communicate with the TA when they need to send messages such as registration, access-blocking, debris on road, and so forth. In return, the TA sends back appropriate confirmation, blocking and issue resolution messages to the RSU. The RSU also sends traffic events to the TA for storing in an incident database. The TA also sends data from the driver behaviour profile when asked for by an RSU attempting to resolve a dispute. An RSU can exchange any kind of WSM and beacon message with other RSUs. Official vehicles can broadcast additional high-priority free road messages towards nearby RSUs to obtain data on free roads while approaching an emergency event.

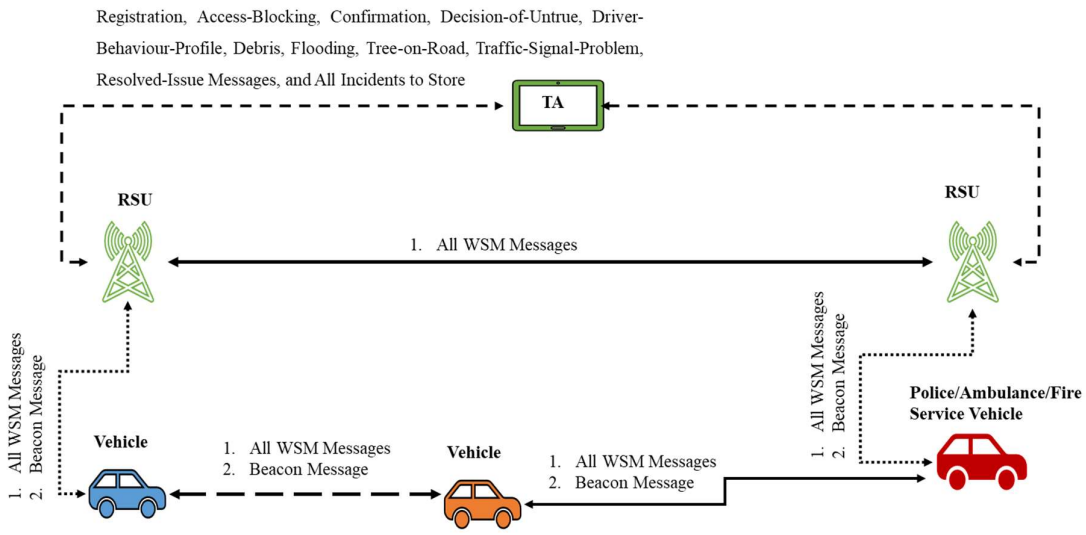


Fig. 3-5: Interaction Among the Entities in the Proposed Trust Framework

3.11 Flowcharts of the Proposed Trust Management Framework

In this section, the flowcharts for all entities are presented. The flowcharts are designed considering their interaction and responsibilities. It should be noted here that a function is only added to an entity when it is responsible for performing it.

3.11.1 Regular Vehicles and the Tamper-Proof Device

Fig. 3-6 is the block diagram of regular vehicles showing the core activities they perform. There are typically three types of actions that trigger a vehicle to perform some activities based on an event. We assume all regular vehicles are communication-ready by following some guidelines at the INIT step, for example, installing a TPD and a dashboard having all the functionalities. Regular vehicles can send and receive announcements and relay them to others. When a user notices an event, then he / she selects it from the dashboard to send in the VANET. The details of the operation are shown in Fig. 3-7 and Fig. 3-8. Another primary action is executed when regular vehicles receive messages. A regular vehicle retransmits a message until its hop count is reached. The associated operations are detailed in Fig, 3-9, Fig. 3-10, and Fig. 3-11. Besides this, some local timers are employed regularly or when certain conditions are met. For example, the application layer of the regular vehicle maintains a timer for generating periodic beacons. A message is scheduled using a timer and when it expires, the vehicle sends the message. Timer operations are depicted in Fig. 3-12.

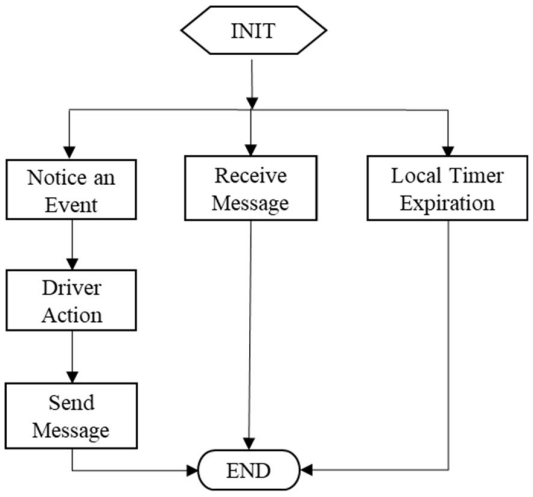


Fig. 3-6: Block Diagram of Regular Vehicle Functionalities

The framework assumes traffic announcements primarily come from a driver's interaction with the dashboard. The class of events on the dashboard is updated according to the trust score of the driver. When a driver is announcing an event, his/her trust score is checked before permitting the announcement. It should be noted that the driver cannot press a button to send an event if his / her trust score is lower than the class of events that contains the observed event because the event will not be present on the dashboard. In this way, the framework achieves access control during message announcements. If the driver has a sufficient trust score to send the message, then the application prepares a message containing the event location, timestamp, vehicle identity, and driver identity and then announces it after pressing the button. The left branch of the flowchart in Fig. 3-7 has a decision module that contains most of the traffic events the framework considers. Additionally, the vehicle application sends metrics to the TPD to calculate the reward/punishment. The right-side branch of the flowchart in Fig. 3-7 activates when the vehicle receives an earlier event and visits the said location within a specific period. If the driver does not notice any symptoms of the event on the road, then he/she may report it by issuing an untrue attack report towards a nearby RSU. The metrics associated with the report are also sent to the driver’s TPD to receive a reward. Fig. 3-7 depicts the flowchart for sending a message from a regular vehicle.

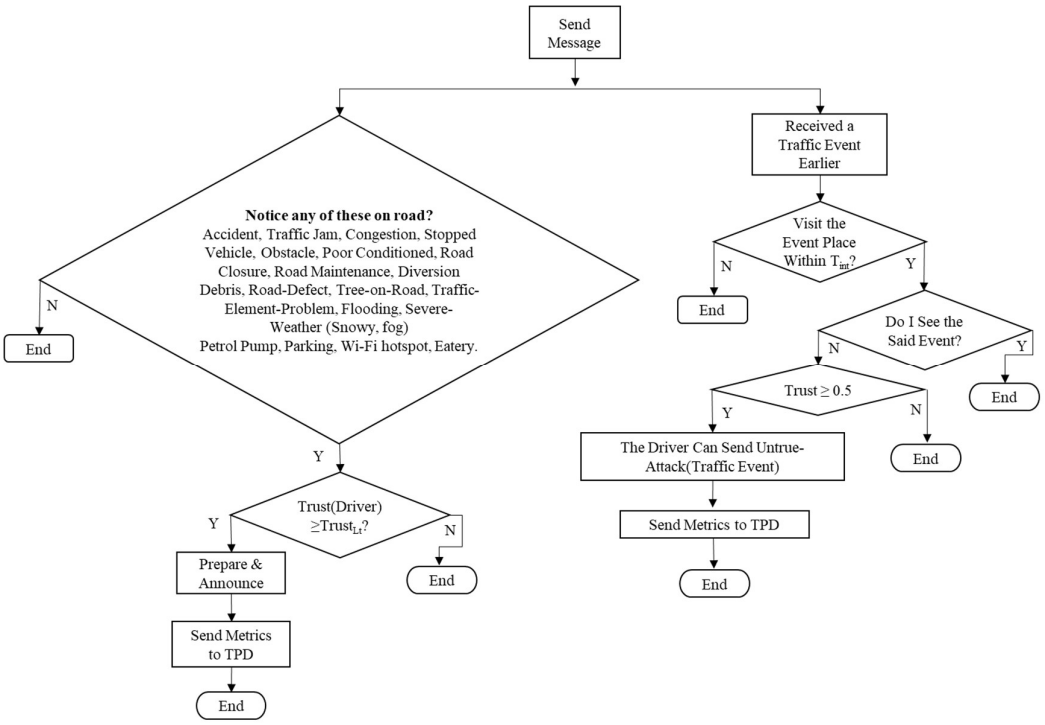


Fig. 3-7: Flowchart for Sending a Message

Fig. 3-8 depicts the trust score-based message announcement capability of a driver which is not shown in Fig. 3-7. Messages are divided into two main groups. Events from the first decision module can be announced when the trust of the driver is greater than 0.5 whereas, the events from the second module can be announced whenever a driver's trust is greater than 0.26. In both cases, the vehicle application prepares the message and can retransmit it up to a configurable number of times.

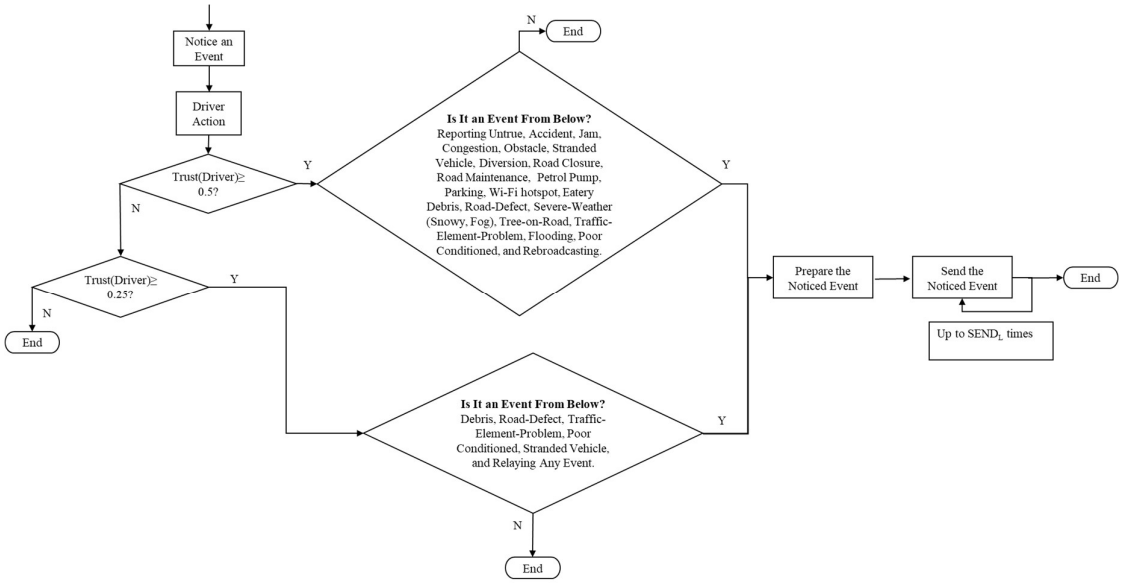


Fig. 3-8: Flowchart for Controlled Message Announcement

When a message is received, the vehicle application has many options as shown in the multiple branches in Fig. 3-9. Whenever an RSU query is received, the application retransmits the message if the hop limit is not reached. To become a clarifier, the vehicle application checks whether the driver is neither the event source nor the reporter of the original event. If the previous check is true, then the message is ignored by the vehicle application. Otherwise, the application checks whether the vehicle has visited the event place within the configurable time and whether the trust score of the driver is sufficient to generate feedback. The feedback containing the view of the driver to the reported attack is prepared and then sent to the queried RSU. After sending a report, the driver sends a notification to his / her TPD to add the reward. If the trust of the driver is not greater than 0.5, the vehicle application cannot generate feedback. The next branch executes when a driver receives an RSU reward or punishment message. Then the vehicle application checks if the message is destined to it to forward the message to the TPD to adjust the trust immediately. The next branch runs whenever a traffic event is received. The vehicle application first checks if it is a new message to store and retransmit. To retransmit an event, the driver must have a sufficient trust score to permit this operation. Additionally, metrics are sent to the TPD for the trust score updating from relaying. The vehicle application also monitors whether the vehicle visits the mentioned location within a specific time interval and the driver notices the event. If the driver suspects the event has not occurred, then he / she may or may not send an untrue attack notification if their trust score is sufficient. If a vehicle receives a beacon message, then it is informed about the sender's vehicle location and can additionally find the distance.

If the vehicle application receives an Access-Blocking message, then it checks the source. If it is from the TPD, then the vehicle sends it to a nearby RSU multiple times to reach the TA via one of the RSUs. Alternatively, if it is from the TA and it is a new message for this vehicle, it means that it is an Access-Blocking confirmation for one of the drivers who drive this vehicle. Then the vehicle application forwards it to the TPD to execute the access-blocking for that driver. If the application finds it is not for this vehicle, then it tests the current hop limit to relay further in the VANET. The flowcharts in Fig. 3-9 and Fig. 3-10 show these activities for regular vehicles and Fig. 3-11 shows some additional activities.

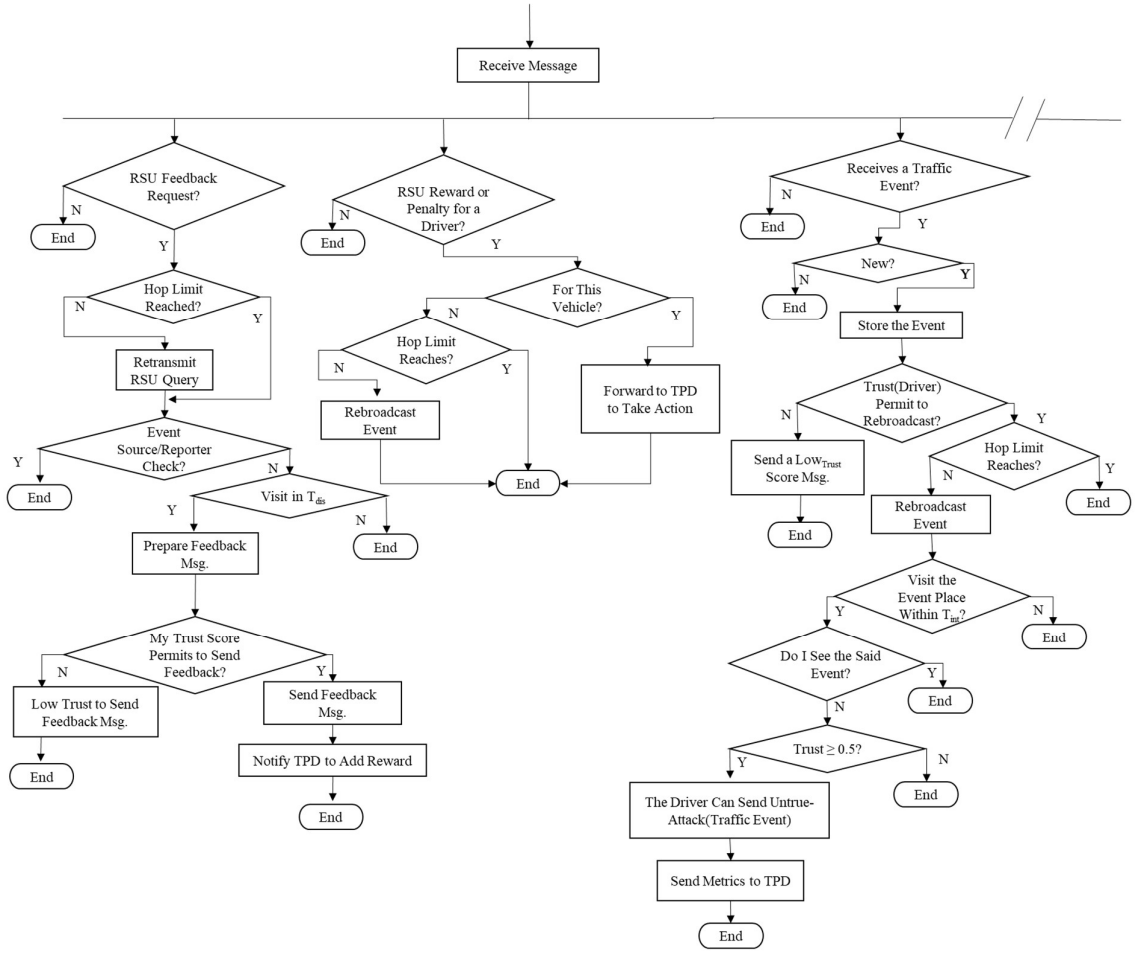


Fig. 3-9: Flowchart for Receiving Messages of Regular Vehicles

The flowchart in Fig. 3-11 shows the activities only for receiving a confirmation of registration or access-blocking and untrue attack from a reporter vehicle. If a regular vehicle receives a registration confirmation message, then it checks if it is the vehicle for which the message is intended. In this case, this message is forwarded to the TPD to let it know that the driver / vehicle is registered in the network. If the message is for a different vehicle, then the vehicle application checks the trust of the current driver and the hop count to relay the message. If the vehicle receives an untrue attack from another vehicle, it checks whether the message is generated from it as the relaying of a message may arrive back at the sender from the relaying of other vehicles. If this is not the case, then it checks the trust and hop count before relaying it again. Otherwise, it ignores the message.

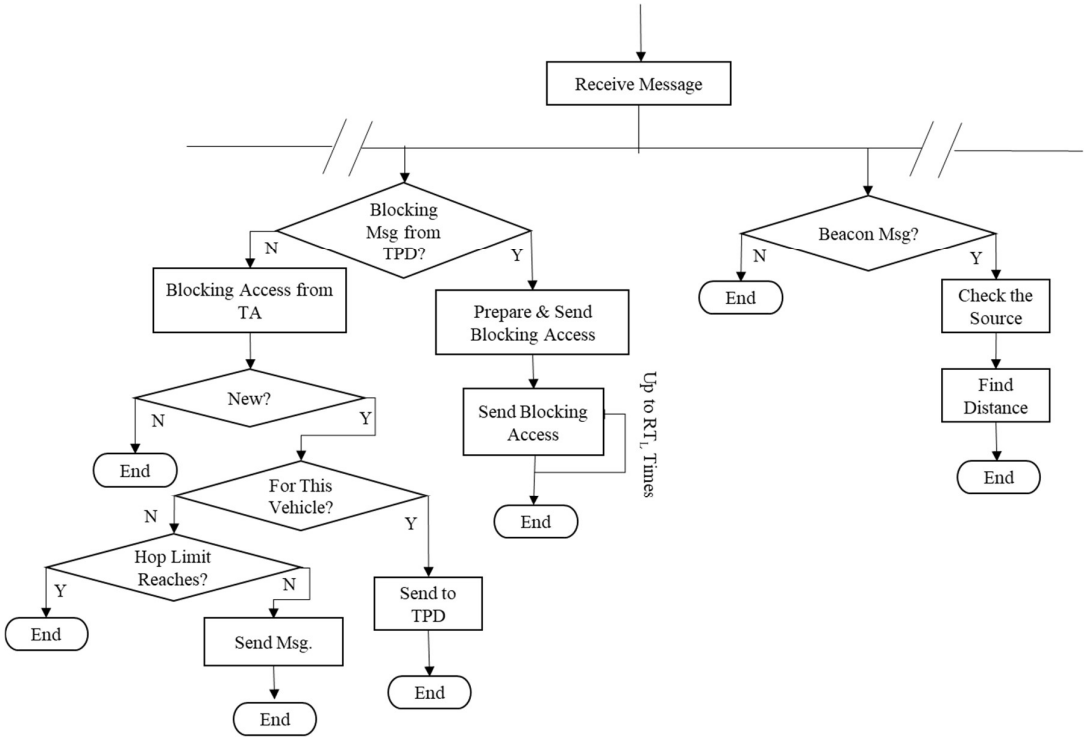


Fig. 3-10: Flowchart for Receiving Messages of Regular Vehicles (Continued)

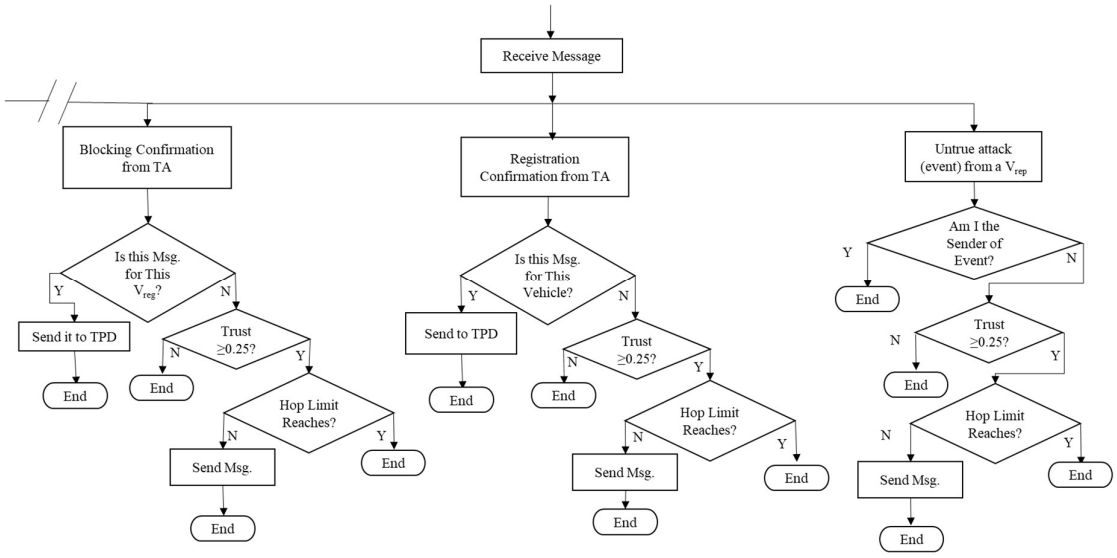


Fig. 3-11: Flowchart for Receiving Messages of Regular Vehicles (Continued)

The flowchart in Fig. 3-12 shows the local timers that a regular vehicle maintains. The first one is used for periodic beacon announcements. When it expires, the vehicle checks the trust score of the driver is at least 0.05 to prepare a beacon message containing the location, speed, heading, and so on. Then it sends the message followed by resetting the beacon timer and this process repeats. Another timer is used when retransmission of a message is required. When the timer expires, the application

checks whether the retransmission limit is reached, and if not, then it retransmits the message followed by incrementing the counter and resetting the timer.

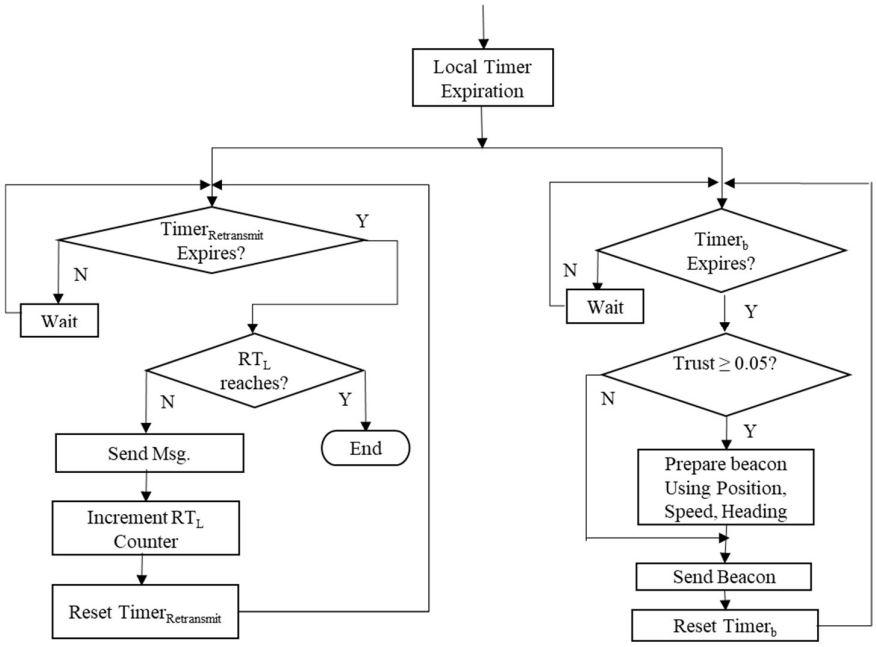


Fig. 3-12: Flowchart of the Local Timers at Regular Vehicles

Inside the TPD, there are three possible courses of action, and each action is event driven. First, assume the TPD is prepared and loaded with the necessary resources at the INIT step. Then the possible actions are sending or receiving messages or local timer expiration. The send message action is triggered whenever a message is sent from the TPD to the vehicle application. Alternatively, the receive message action is executed whenever the TPD receives a message from the vehicle application or any external source. The TPD maintains two local timers for periodically checking the condition of access-blocking and for reward-withholding. Fig. 3-13 depicts a block diagram of the TPD primary actions. There is a dedicated flowchart for each primary action which will be explored subsequently.

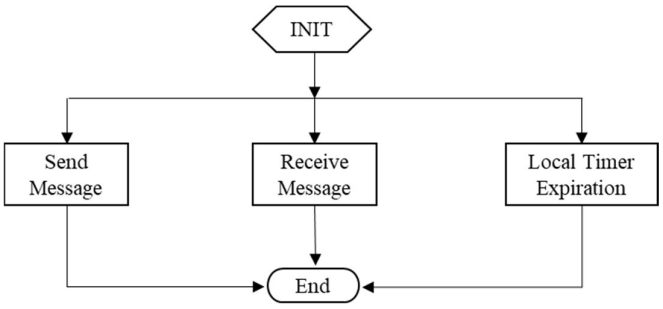


Fig. 3-13: Flowchart Depicts the Primary Actions for a TPD

The TPD can receive multiple types of events from the vehicle application. First, the TPD can receive a complaint about an earlier announcement since the event is reported by a reporter. Second, it can receive metrics information via the vehicle application from the announcements. Then, the TPD applies a set of rules to determine the amount of the reward / punishment for an announcement. After that, it starts a reward withhold timer in case it is a rewarding action, otherwise, for a punitive action the driver receives the punishment instantly. Third, whenever the TPD receives an RSU reward / punishment, the TPD first checks whether it is already added, and then it skips the trust adjustment in case of repeated reception. Otherwise, the driver receives the reward / punishment from the RSU instantly. In both cases, the trust of the driver is kept within the limit (0.05 to 0.9). Fourth, whenever a driver change request is obtained from the wireless card reader, then the TPD checks whether the requested driver is new to this TPD or already known. If the driver is known to the TPD, then he/she resumes driving with the previous trust score, whereas, for a new driver, the TPD gives him/her an initial trust score if he / she is using the VANET for the first time. In the case of a driver who has used a different vehicle previously, the TPD issues a trust query for a driver which the vehicle application broadcasts, and this message reaches the TA via an RSU. When the vehicle receives a trust update from the TA, the TPD sets the trust score using the value from the TA. A driver can send a trust update message to a nearby RSU to relay it to the TA to store the trust record of the current driver. Fifth, if there is a relaying event notification from the vehicle application, the TDP updates the trust immediately with a relaying reward when trust of the driver / vehicle is less than 0.5. Similarly, the driver receives beaconing rewards if his / her trust score is less than 0.5. Lastly, if there is a blocking confirmation from the TA, the TPD executes the access-blocking action for the concerned driver by disabling the application and/or transmission. Fig. 3-14 depicts these interactions using multiple branches in the flowchart.

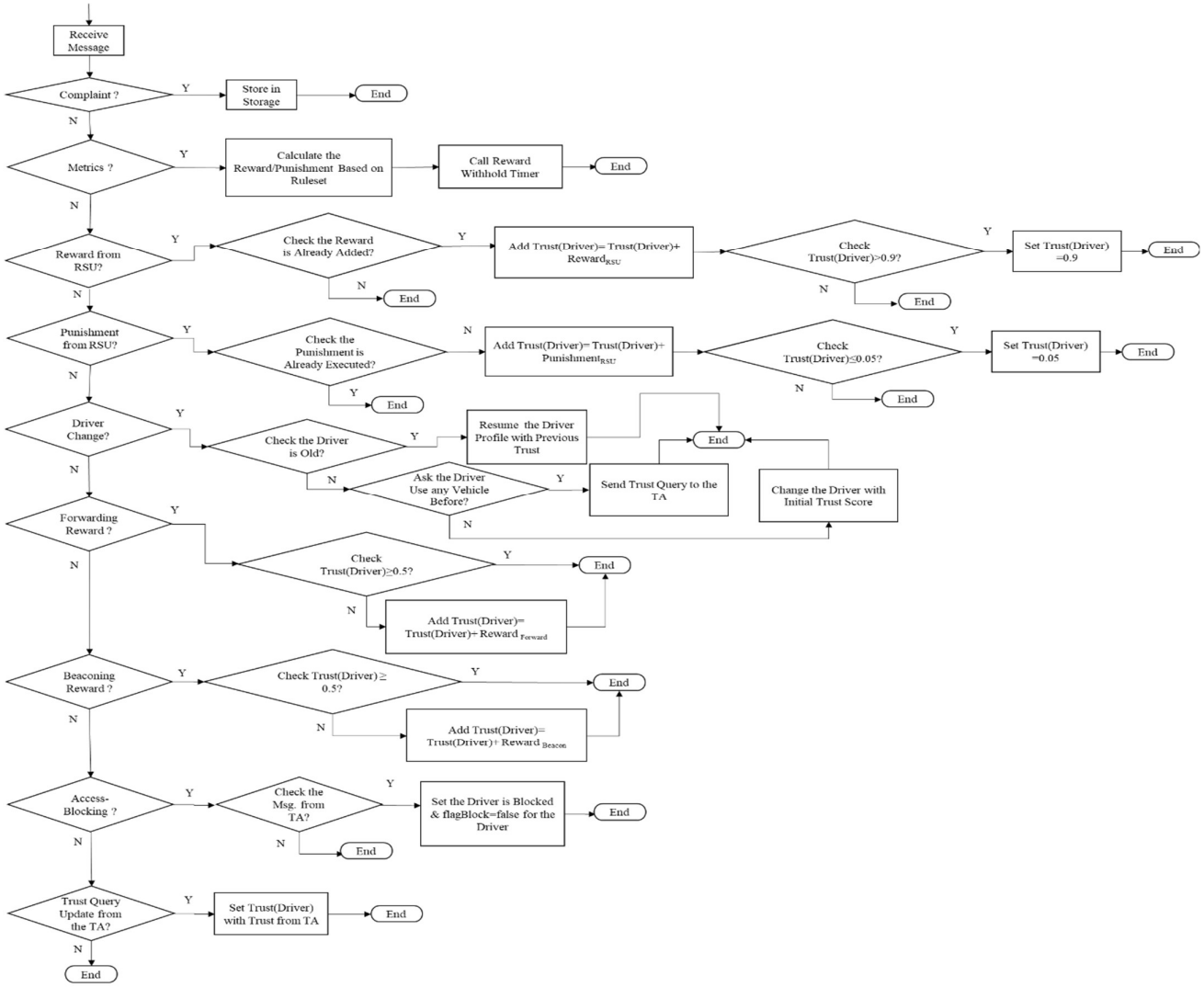


Fig. 3-14: Flowchart for Message Reception at a TPD

The send message action is executed whenever the TPD sends an access-blocking request message to the TA or sends a warning message to the current driver, for example, saying ActHonestly otherwise, your transmissions will soon be blocked. The latter is possible as the TPD saves the trust of drivers in its database. The flowchart in Fig. 3-15 depicts the send message activities of a TPD inside regular vehicles.

The TPD maintains two local timers. The first one is the reward withhold timer which is used for withholding the reward for an announcement for a period. During this period, the TPD waits for any complaint raised by any reporter within the framework. If this is the case, then the TPD does not add the reward when the timer expires. On the other hand, when there is no complaint, upon the timer expiration the TPD of the sender vehicle adds the reserve reward. It should be noted that for punitive action, the TPD subtract the punishment from the current trust of the driver immediately without starting a timer.

There is another timer inside the TPD to check whether the blocking access condition is met or not. If the condition is met, then the TPD prepares and sends a blocking access request message to the TA via the vehicle application. Also, the timer is reset based on the trust score of the driver. For example, if the trust is greater than 0.05 and less than 0.25, then the timer is set to 60 seconds. This duration should be short compared to higher trusted vehicles because vehicles with this trust score need to be checked more frequently for access-blocking otherwise, they will get opportunity to announce more untrue messages, due to dispute resolution process needing some verification time. Thus, they need to be checked quickly whether any vehicle / driver meets the access blocking condition. When the trust is in the range $0.25 \leq \text{trust} < 0.5$, then the period of the timer is set to 120s. This duration is set higher than the previous scenario as vehicles have more trust than the previous case. Thus, their access-blocking condition is called less frequently. Next, when the trust is greater than or equal to 0.5 but less than 0.8, then the timer period is set to 180 seconds. This should be longer than the previous two scenarios as vehicles remain trustworthy and hence the access-blocking condition can be checked less frequently than the former two cases. Finally, the timer duration is set to a longer value, for example, 300s for highly trusted vehicles / drivers which has trust score of 0.8 to 0.9. These durations are smaller (i.e., 1 minute to 2 minutes) than the verification time for low trusted drivers to limit the scope of malicious announcements prior to blocking. Alternatively, these durations are higher (i.e., 3 minutes to 5 minutes) for trusted drivers as access blocking is not needed when they maintain good trust score. However, the RTA can change these values as needed. The duration of the blacklist timer can be configured to different values as well. These two timers are depicted in Fig. 3-16.

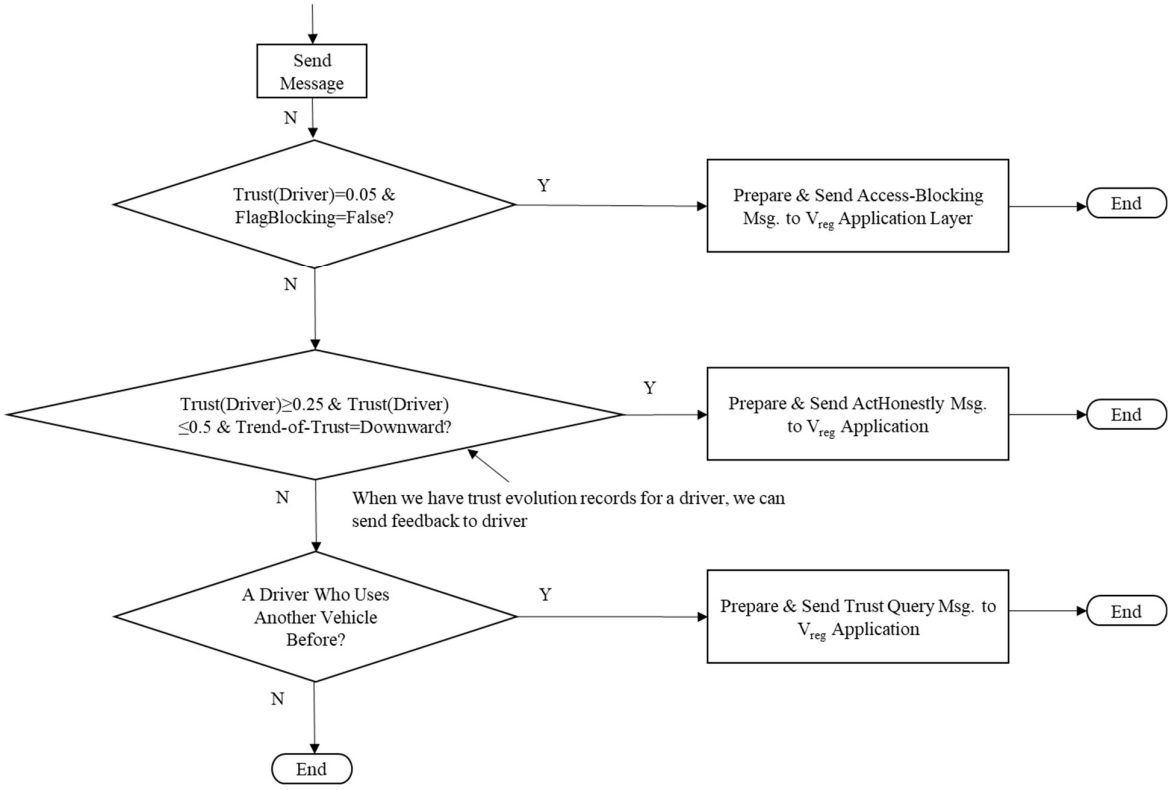


Fig. 3-15: Flowchart for Message Send Activities for a TPD

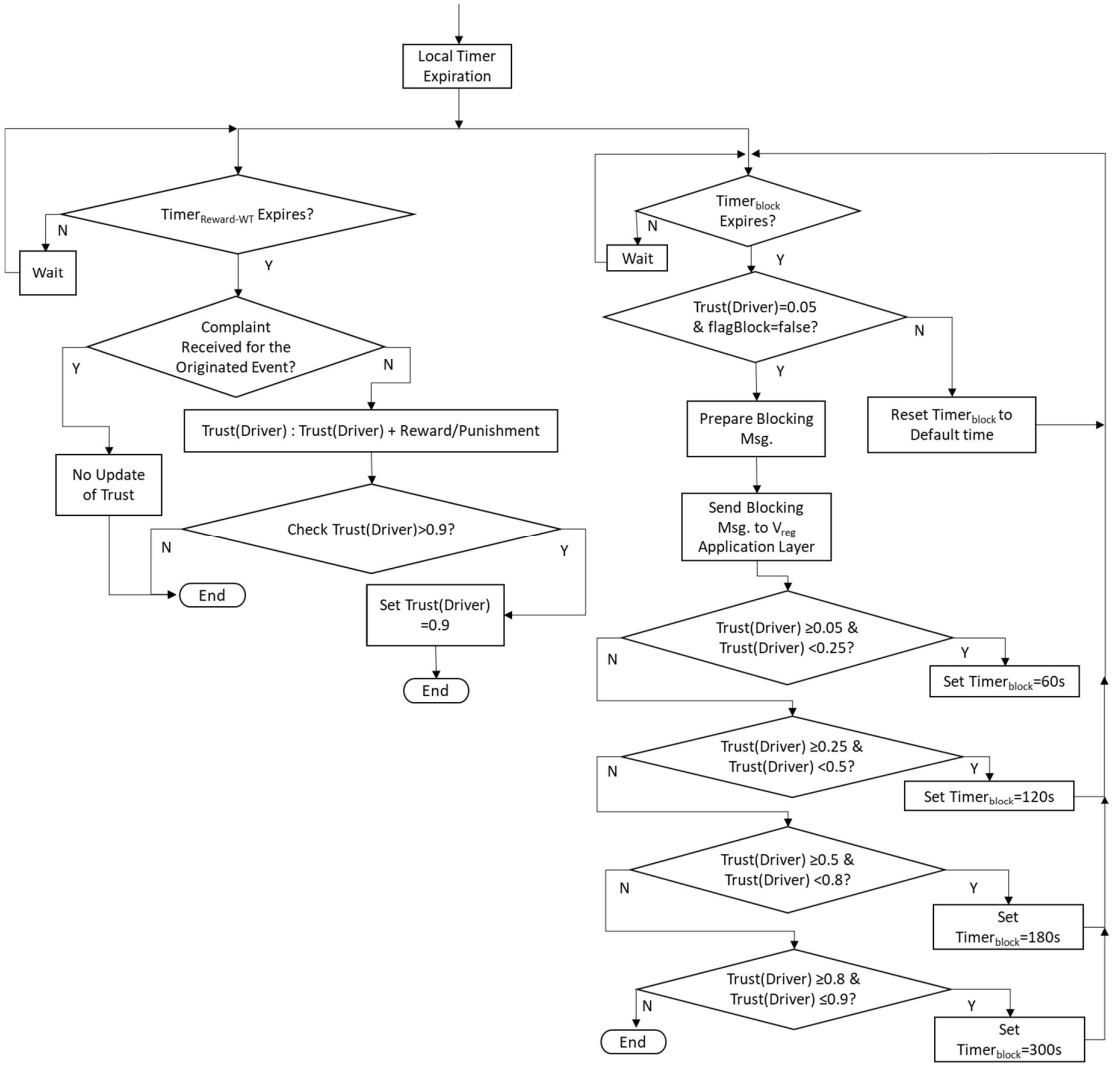


Fig. 3-16: Flowchart for Local Timers at the TPD

3.11.2 Official Vehicles

We assume the official vehicle application is loaded and initialized at the INIT step. After that, the vehicle can send or receive messages through this application. A branch is executed only when there is a corresponding event. The periodic beacon is transmitted from the application unit which is a timer-driven message containing vehicle status information. The official vehicle sends messages to other entities, as needed. Alternatively, the message receive action is executed whenever an official vehicle receives a message from another entity. Fig. 3-17 shows these primary actions in a flowchart. There is a separate flowchart for each activity type from the flowchart in Fig. 3-17.

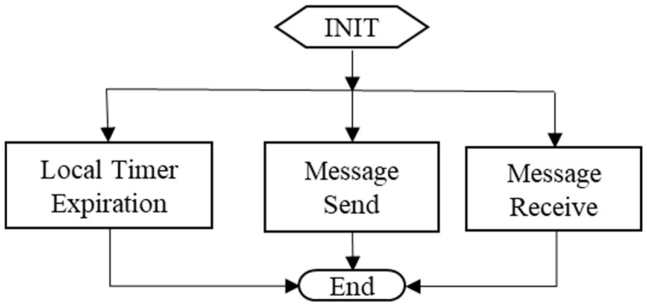


Fig. 3-17: Flowchart Depicts the Three Main Actions for Official Vehicles

The official vehicle unit can also announce an observed event on road, or it can report an event from another vehicle. To report an event, the official vehicle needs to visit the event place within a specific period and needs to locate or find the symptom of the event on the road. Fig. 3-18 shows the message announcement activities from an official vehicle.

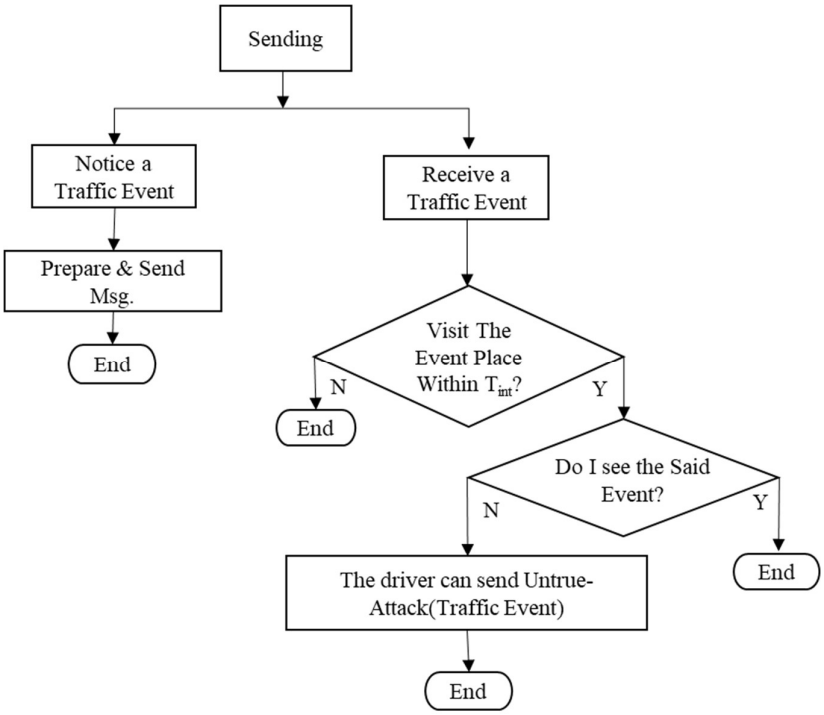


Fig. 3-18: The Flowchart for Sending a Message

Every official vehicle takes an appropriate action based on the type of received message similar to that for regular vehicles in most cases. In some instances, official vehicles generate authoritative high-priority messages to inform others about their approach towards an emergency event on road. In these cases, regular vehicles should free the lane concerned so that the official vehicle can reach the event location without delay. Also, this message is important to an RSU as they are informed about an official vehicle approaching a severe event. Whenever a new event message arrives, the official vehicle first

checks whether it is receiving it for the first time, otherwise, it ignores repeated reception. If this test proceeds, then the vehicle checks whether the event is severe or not. For example, severe events are accidents, heavy traffic jams, floods, or snow / ice on the road which need extra scrutiny. If this is the case, the nearby official vehicle sends an addressing message and a free-road message several times. The addressing message informs the nearby RSU that an official vehicle is visiting the event location. Finally, when the issue is resolved, the official vehicle generates a road-clear message in the neighbourhood which may reach one or more RSUs. The official vehicle may receive a feedback query message from a nearby RSU to give an opinion about a reported event. When it is a new query, it checks whether the hop limit of the message is reached for relaying. If it is not, then it checks whether the official vehicle has visited the event location within a certain period. If this is the case, then it prepares the feedback message to send and retransmit it up to configurable times. However, if the vehicle has not visited the event place within a certain time, then it sends a far-from-the-place message to the querying RSU, and this message is also retransmitted multiple times to reach the RSU. Fig. 3-19 shows a flowchart for the set of activities executed at an official vehicle for a received message.

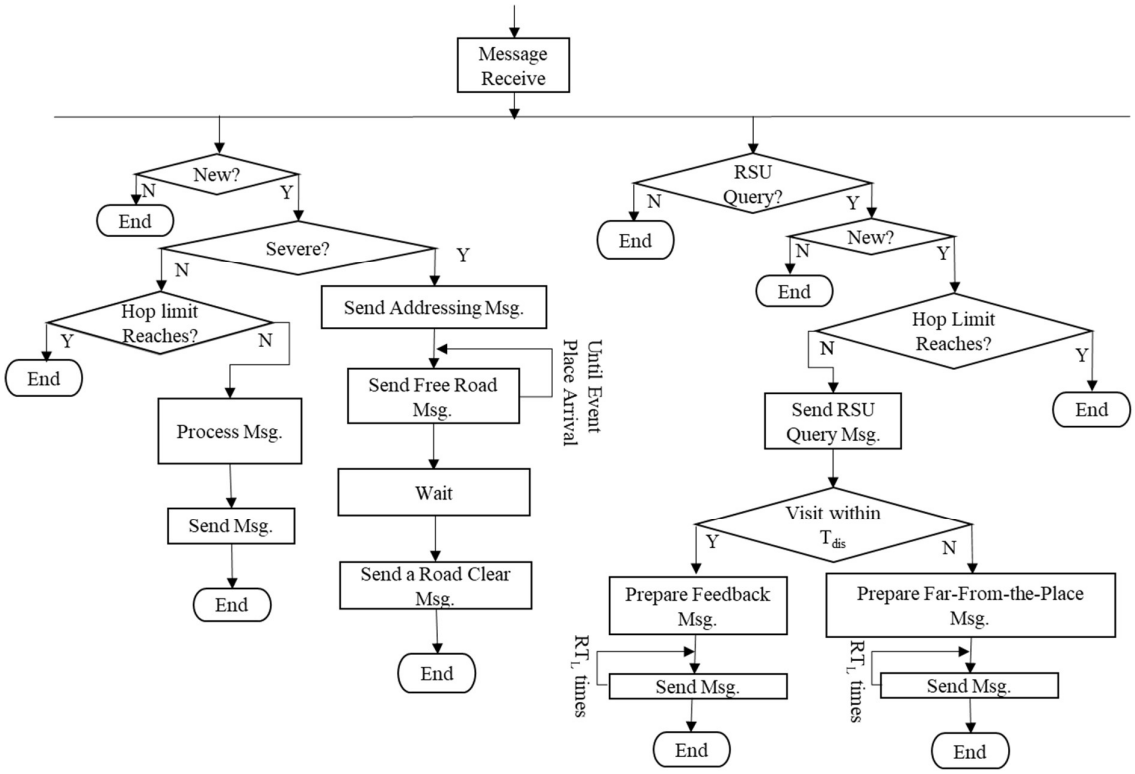


Fig. 3-19: Flowchart for Message Reception at an Official Vehicle

3.11.3 Road-Side Units

The primary actions for RSUs are depicted in Fig. 3-20. We assume the RSU application is initialized with its resources at the INIT step. An RSU employs local timers for periodic event handling. For example, a timer periodically triggers the sending of RSU beacons. An RSU can receive messages from

different entities and their action is different based on the type of message source. Hence, we present different flowcharts which are named as flow1 (TA), flow2 (RSU), flow3 (regular vehicles), and flow4 (official vehicles) for received messages from each source type as shown in Fig. 3-21. RSUs send traffic updates of events periodically to the vehicles by retransmitting the same message until the event is resolved. RSUs also communicate with nearby RSUs and the TA.

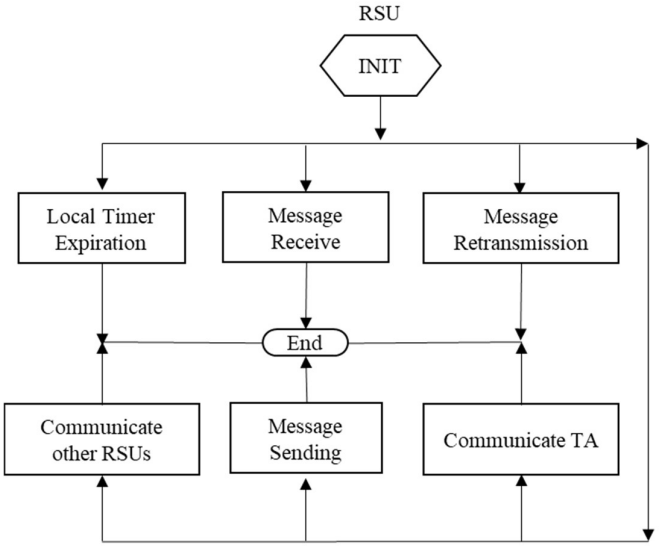


Fig. 3-20: The Block Diagram View of RSU Functionalities

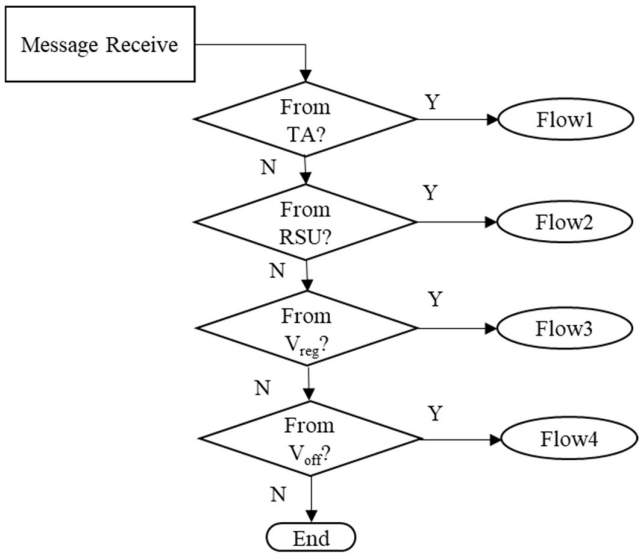


Fig. 3-21: RSU Receive Message Classification from Different Sources

An RSU manages two different timers for beacon management. When one timer expires, an RSU generates a beacon which is followed by resetting the timer, and this process is repeated. An RSU also maintains another timer which determines when to delete the old received beacons as the RSU stores

beacons for a given period to facilitate authoritative verification by identifying nearby vehicles. Another timer is used with dispute resolution. When it expires, an RSU decides using the data collected during the period and sends reward / punishment to the corresponding entity. The RSU informs the TA of its decision as well as the nearby RSUs. The flowchart is in Fig. 3-22. shows the local timers an RSU maintains.

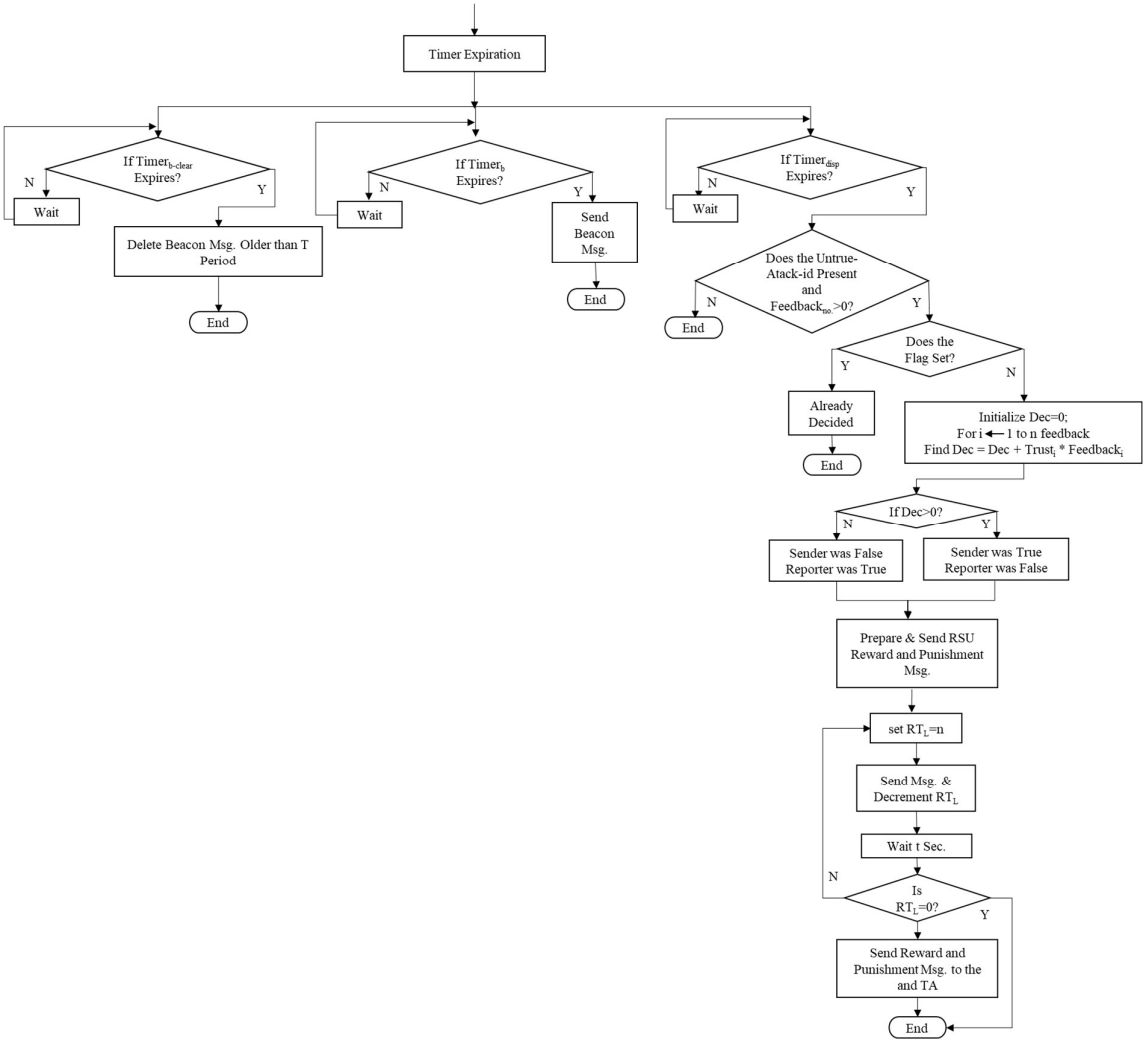


Fig. 3-22: The Flowchart for the Local Timers at an RSU

When an RSU receives a registration confirmation message for the first time from the TA, it checks whether it is for a vehicle, or for itself. If it is for a vehicle, then it retransmits the message up to a configurable number of times. If it is for itself, then its registration is confirmed. Whenever an access-blocking confirmation message is received from the TA, an RSU checks whether it is a new message to announce to a vehicle. First, it saves the message into its storage and then retransmits the message multiple times. The message can be a road issue resolved message from the TA. In this case, if it is a new message it saves the message and retransmits it multiple times. Otherwise, it ignores the message.

In the case of a driver behaviour profile message from the TA, the RSU processes the message to retrieve the desired metrics and then uses the metrics for the required purpose. The flowchart in Fig. 3-23 shows the possible activities when a message is received from the TA.

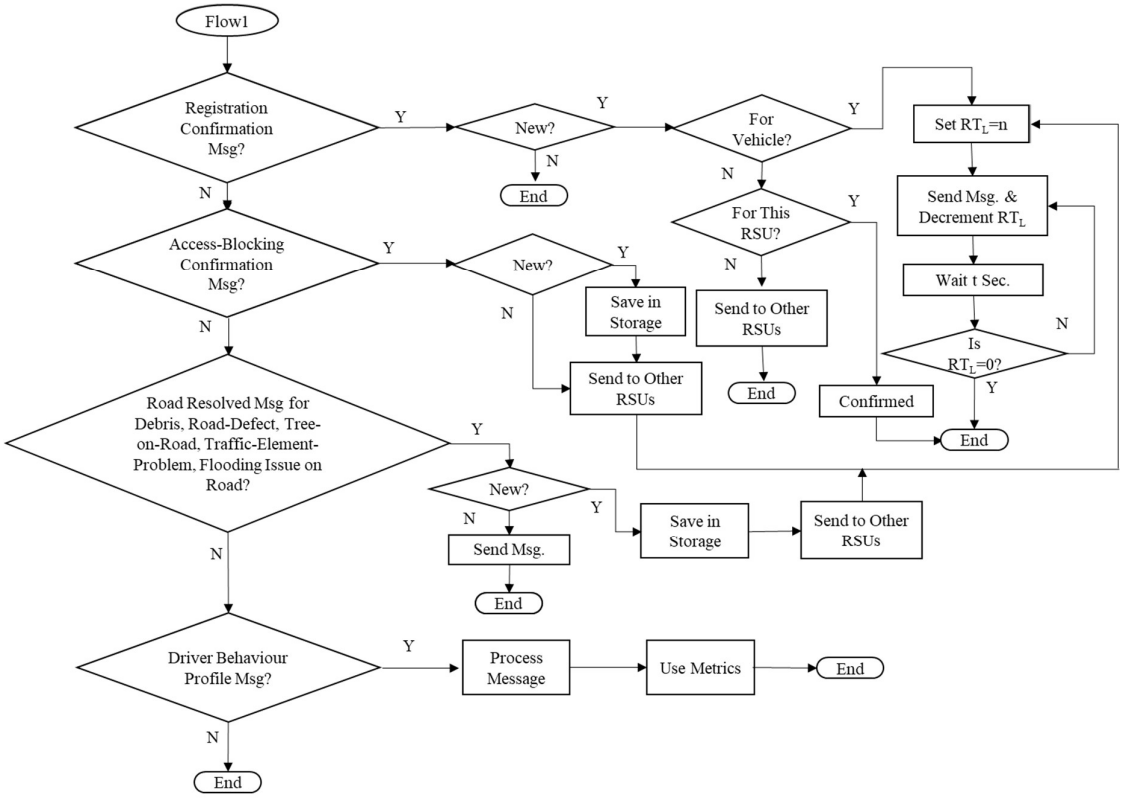


Fig. 3-23: The Set of Actions for Messages Received from the TA

The flowchart in Fig. 3-24 depicts the reception of messages from the neighbouring RSUs. An RSU performs different activities based on the message source. Also, messages are grouped according to similar activities performed at an RSU. First, if an official vehicle generates a far-from-an-event message when it is not near to the event location, then the RSU checks if it is a first-time reception and so stores it; otherwise, it discards the repeated reception. Secondly, when an RSU receives a "free road" message from an official vehicle, it checks whether the severe road condition is resolved or not. If it is not resolved, then RSU saves the message and prepares a "restricted movement on a road" message towards other vehicles so that the official vehicle can experience a road free approach. If the RSU receives a service reply from another RSU, it checks if the message is a new one; then it stores and announces it up to a configurable number of times to vehicles. If it is a resolved issue message, it checks if the message is a new one; then it saves it into storage and announces it to vehicles up to configurable times. If the message is a new traffic jam or congestion message, then the message is saved first and announced until a "road-sorted" message is received. For the reception of a new service query, the RSU sends a reply up to a fixed number of times. If the message is a poor-condition road, a diversion message

or similar, the RSU stores it, if new; otherwise, it discards the message. The reward or punishment message from a dispute decision is stored first if it is a new one and then it is announced multiple times; otherwise, the RSU ignores the repeated reception. When an RSU receives a new feedback message from a different RSU, then the RSU checks the issue is listed in it and that is not resolved. If these are true, then it checks whether the message is from an official vehicle to mark the issue as “resolved”. Otherwise, it stores the message.

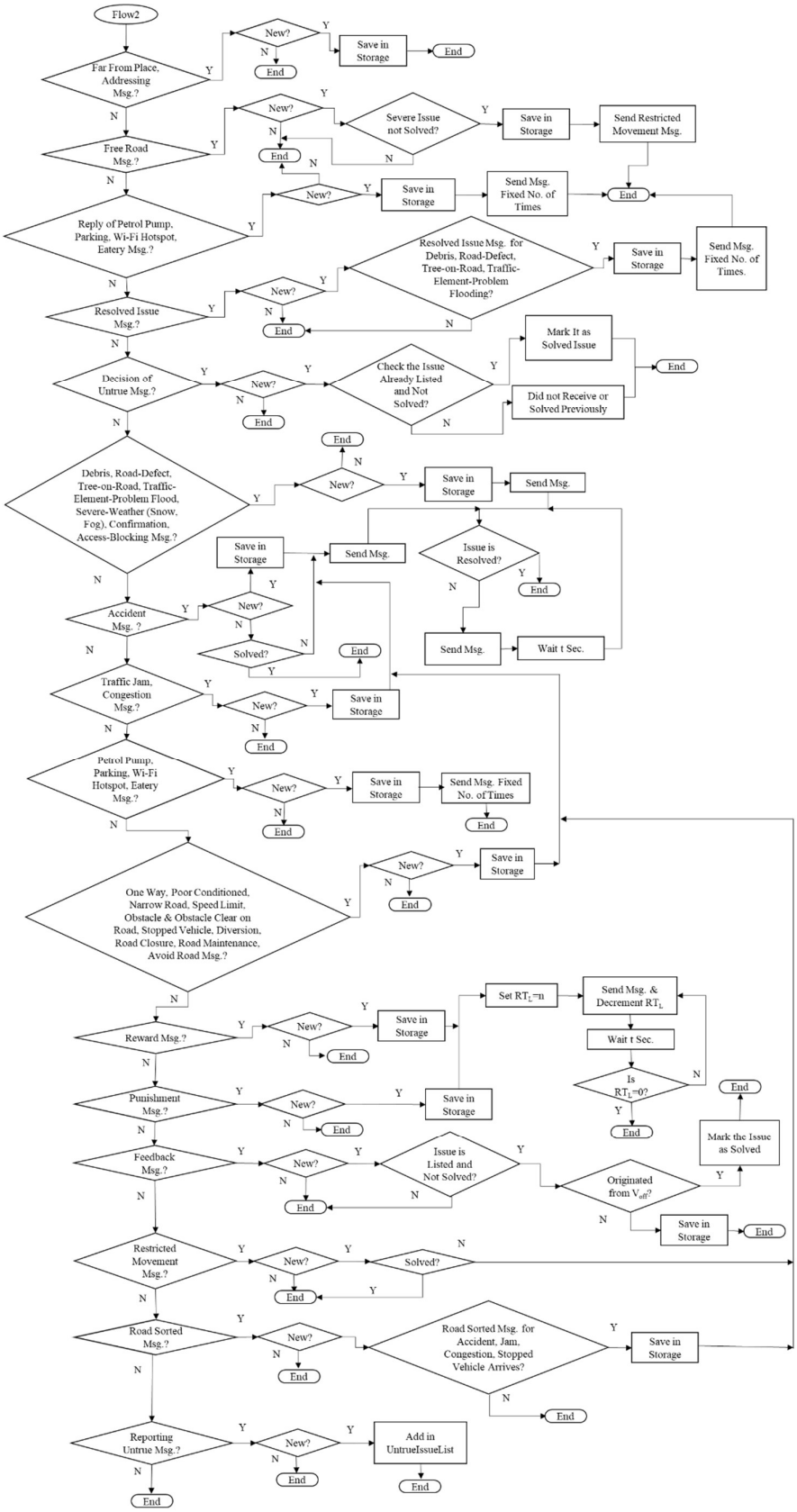


Fig. 3-24: The Flowchart for Message Reception from other RSUs

The flowchart in Fig. 3-25 shows the activities involved with the messages from vehicles. When there is a beacon, the RSU stores it in its local storage. It maintains a vehicle list containing the vehicles that regularly visit its coverage area and sends a regular visit message. An RSU can ask a regular visiting vehicle whether or not it knows about an event with more confidence. If a service reply message from another RSU reaches an RSU, it stores the message if it is a new one and retransmits it if the configurable hop limit is not reached. If there is an accident / traffic jam / congestion / avoid road message, then the RSU stores it besides relaying it to nearby RSUs. The RSU then retransmits this message until the issue is resolved. If there is a stranded vehicle / an obstacle on road, then the RSU repeats announcing the same message until the issue is resolved. If a new diversion/poor-condition road/road closure or maintenance message arrives at an RSU, then it saves it into local storage which is followed by forwarding it to nearby RSUs. After that, it informs the vehicles on the road until the issue is resolved. If there is a new service query from a vehicle, then the RSU prepares a reply either providing the requested information or saying it does not have the requested information. In this case, the framework assumes the RSU only repeats announcing the reply twice. As this service reply only needs a particular vehicle and which may not be required by all. If there is a new "restricted movement" message from a vehicle, after storing the message, the neighbouring RSUs are also informed. Then the RSU checks whether the issue is resolved or not. If the issue is not solved, then it is announced by the RSU multiple times. If the RSU receives a reward / punishment message from another RSU, then it relays the message to reach the concerned drivers.

If there is a feedback message, the RSU checks whether the issue relating to the feedback is listed already or not. If the message is for a dispute which is being considered by this RSU, then it stores the feedback for handling by the dispute resolution process. If the issue is not listed at this RSU, then it records the issue. If there is a report of an untrue attack, then the RSU first checks whether it is already listed or not. If it is not, then the RSU lists the untrue attack first and sends a message to nearby RSUs. After that, it prepares a query message regarding the disputed event. Then, the RSU starts a collaboration timer to collect feedback from clarifiers and official vehicles to decide on the validity of the event. Next, the RSU sends the query message repeatedly up to a specified number of times. If it is a debris / road-defect / tree-on-road / flood / access-blocking message, then the RSU stores it first and then informs nearby RSUs and the TA. If the RSU receives a road-issue resolved message for debris / road defect / tree-on-road / flood, then it stores the message and sends it multiple times.

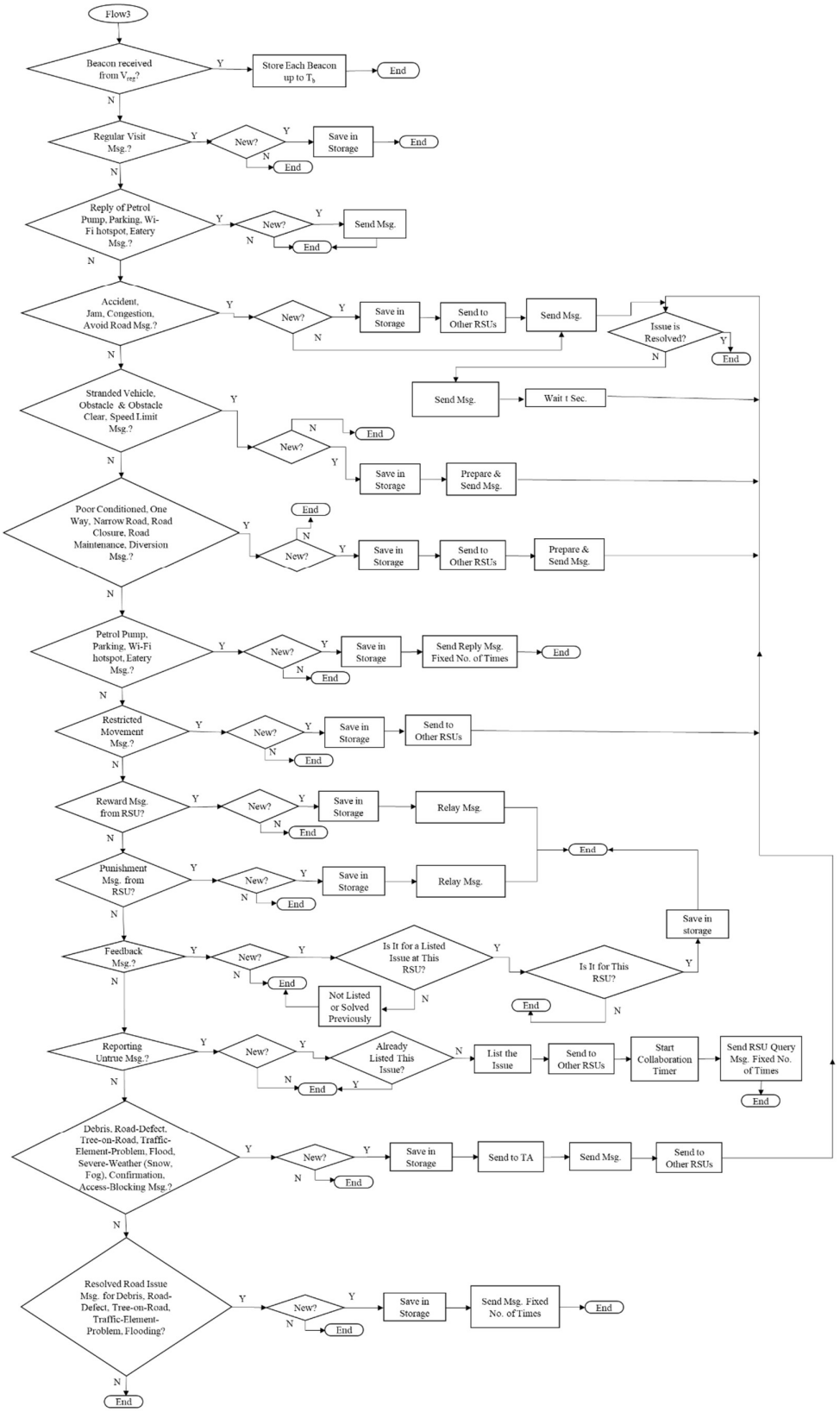


Fig. 3-25: The Flowchart Depicts the Activities for Messages from Regular Vehicles

If the RSU receives an event from the list of messages in the first decision module containing "accident, traffic jam, obstacle & obstacle clear" and so on from an official vehicle, then it performs a similar course of action as it performs for message reception from a regular vehicle. If there is a sorted-road message from an official vehicle, then the RSU checks if the message is for any accident, traffic jam, or congestion. If this is the case, the RSU sends the sorted-road message multiple times. If the RSU receives an untrue attack report from an official vehicle, then it first checks whether the issue is already listed; if not, it lists the issue and then forwards it to nearby RSUs. As this is an untrue attack report it is itself sufficient to decide upon the attack. In this case, the RSU only repeats announcing the punishment message for the sender vehicle multiple times to ensure it receives the RSU punishment. The RSU also sends this message to nearby RSUs to announce themselves as well as to the TA to store it in the driver behaviour profile database. When an RSU receives a feedback message from an official vehicle, it first checks whether the issue is listed and determines if the feedback is for this RSU. If it is for this RSU, then it uses this to decide the outcome of the dispute and then prepares reward and punishment messages which are announced multiple times to reach the concerned drivers. Fig. 3-26 shows a flowchart for message reception from an official vehicle at an RSU.

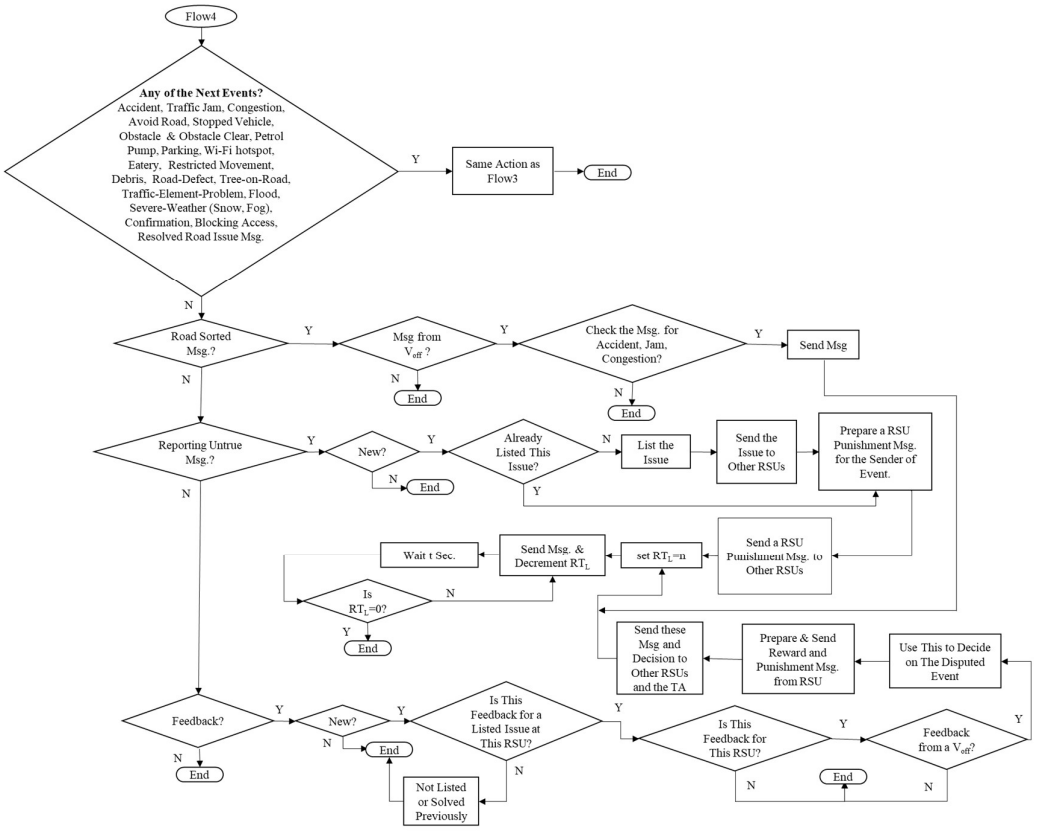


Fig. 3-26: The Flowchart for a Message Reception from an Official Vehicle

An RSU forwards the listed messages in the decision module to nearby RSUs to broadcast themselves as shown in Fig. 3-27. An RSU also communicates with the TA whenever it receives road

issues, registration, or access-blocking messages, and then waits for the confirmation or a resolved issue message for the respective problem. The RSU saves the returned messages from the TA into storage and then sends the message in the VANET up to a configurable number of times. Fig. 3-28. depicts the set of actions an RSU follows with a message from the TA.

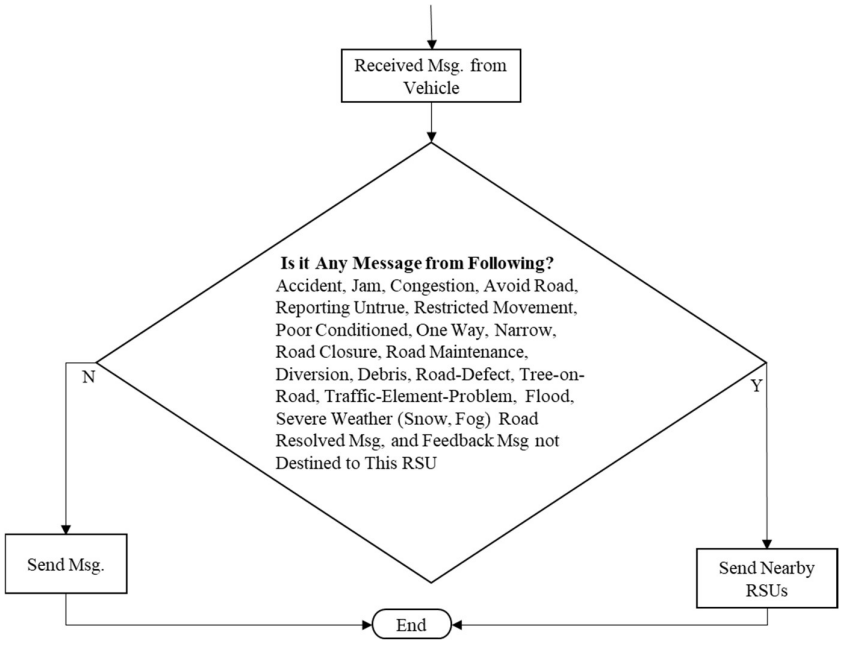


Fig. 3-27: Messages That Need to Send to Nearby RSUs

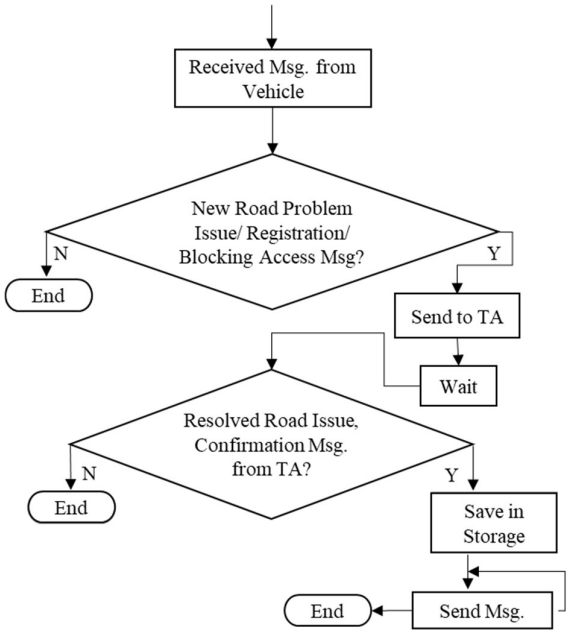


Fig. 3-28: Messages That Need to Send to the TA

The collaboration process starts at an RSU with the reception of a complaint/report/untrue attack message from a reporter's vehicle. However, a report from an official vehicle or another RSU does not trigger the dispute resolution process. The reason behind this is when the message comes from an RSU, then the framework assumes the untrue attack is already being handled by another RSU. Hence, the RSU that receives the disputed event from another RSU lists the event in an unresolved-issue database if it has no information about it, otherwise, it ignores the repeated reception. When a report from an official vehicle is received at an RSU, the RSU directly assigns a punishment to the sender of the untrue event if the RSU is considering the dispute for which the feedback comes. It also announces punishment messages several times. The RSU updates nearby RSUs and the TA about the punishment.

If the new untrue attack report is from a regular vehicle, then the RSU checks whether the disputed event is already resolved or not. If it is not resolved yet, then the RSU stores it in an UntrueIssueList and informs nearby RSUs about the attack. Meanwhile, the RSU prepares a query to collect feedback from vehicles that are visiting the region where the event is announced and reported. Then the RSU starts a timer for feedback collection. For every received feedback, the RSU checks if the source is an official vehicle to decide quickly about the ongoing dispute. Feedback from an official vehicle is used as the decider followed by creating a reward and a punishment message by setting the assessed amount for each driver. Then the RSU informs nearby RSUs and the TA. The RSU announces these messages a few times to reach the appropriate drivers and saves the issue in a resolved issue list. Otherwise, when the timer expires, the RSU multiplies each feedback with the respective trust and then takes the sum of the product, and checks the sum is not zero. If the sum is zero, the issue is still unresolved. Then this RSU waits for a nearby official vehicle to ask it to inspect the region and send a report. When the official vehicle sends feedback, the RSU can use this message to reward or punish the drivers. On the other hand, if the sum is either greater or lower than zero, then the RSU reaches a decision and announces reward and punishment by the same process as mentioned earlier. These actions are shown in Fig. 3-29.

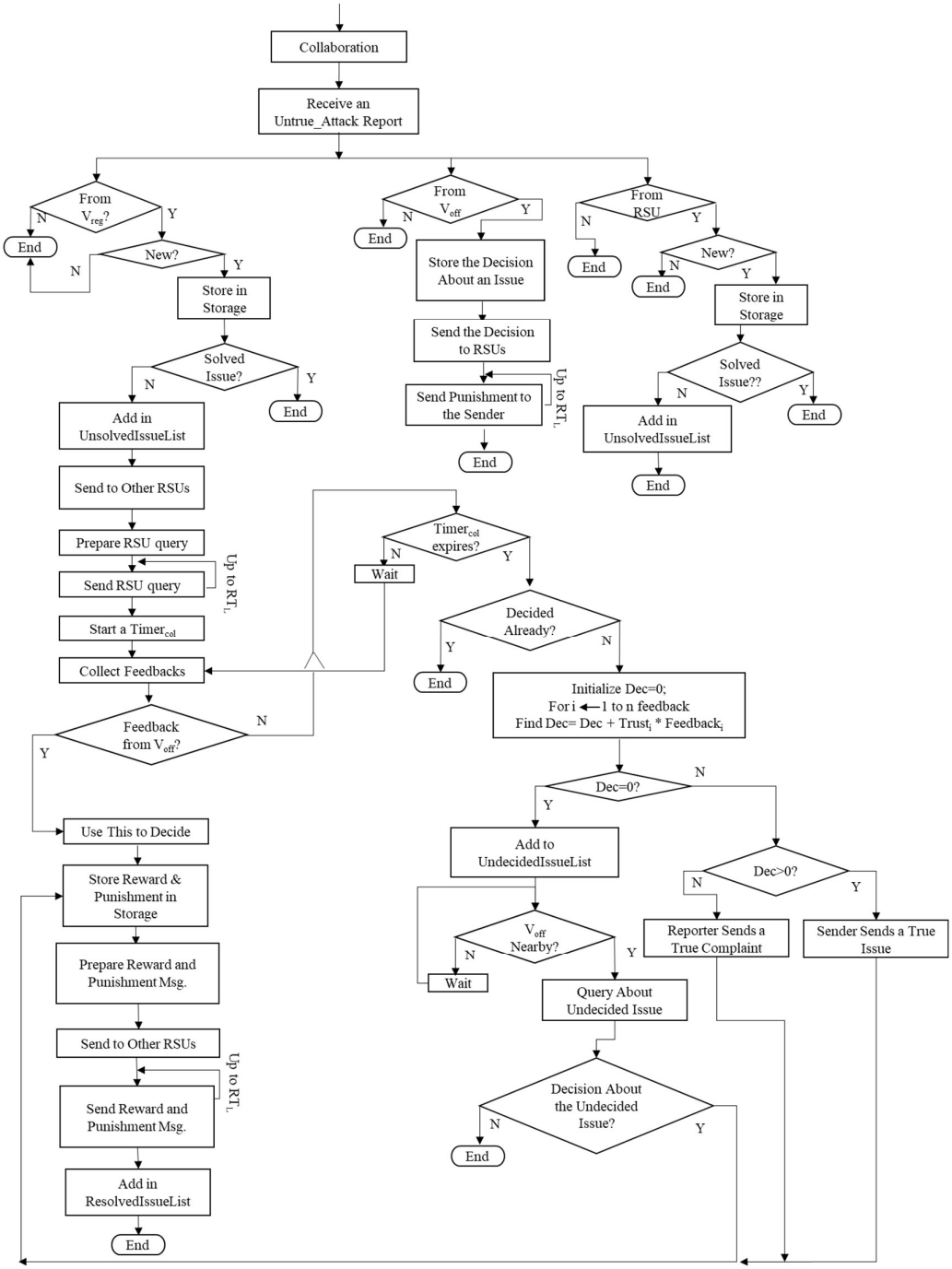


Fig. 3-29: Flowchart for Dispute Resolution Process at an RSU

Fig. 3-30 shows a flowchart for message retransmission by an RSU. When messages come from the TA or an RSU or a regular vehicle they typically need to be retransmitted multiple times. The RSU first checks a message is new and so stores it and announces it multiple times. However, severe messages from a regular vehicle are forwarded to RSUs in the nearby region. For a message from an official vehicle, the RSU first saves it into local storage and then announces it a configurable number of times.

If the event is severe, the RSU also sends a “avoid a specific road” due to a specific reason. If this is not the case, the RSU just waits for a road clear / sorted message from an official vehicle. When a road clear message comes, the RSU retransmits it multiple times to vehicles, so they are aware that the issue is resolved.

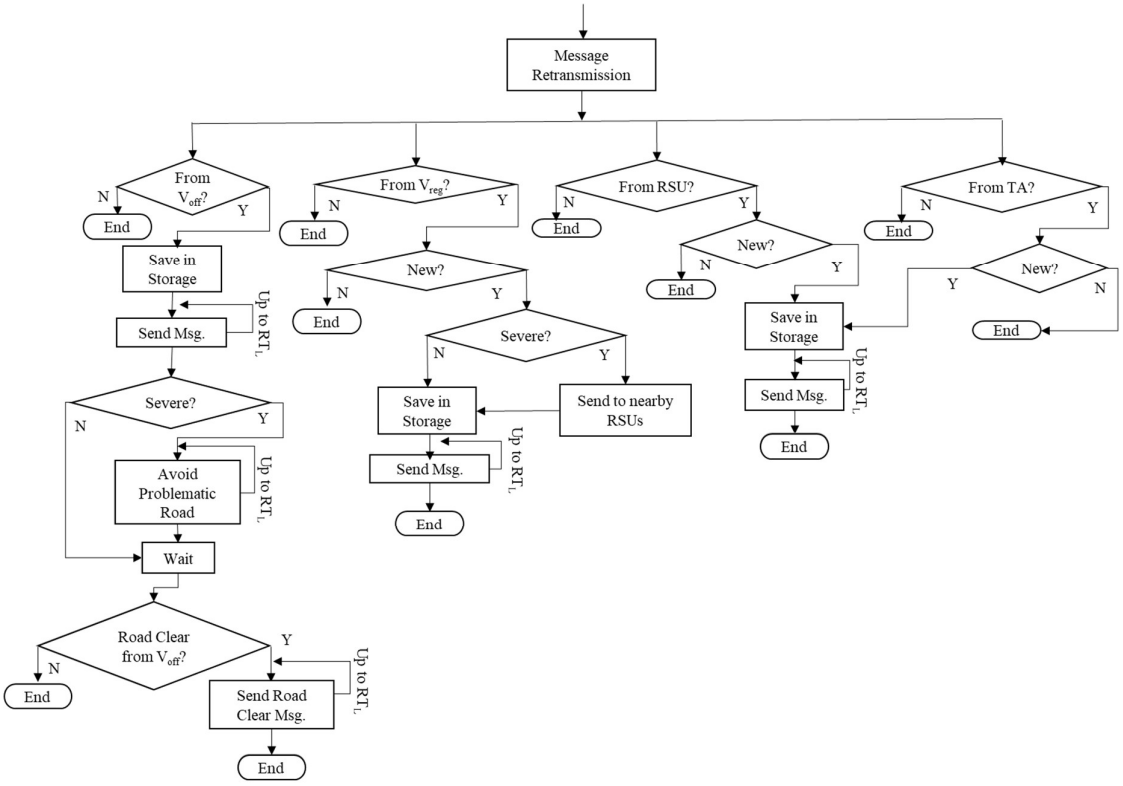


Fig. 3-30: Flowchart for Message Retransmission at an RSU

When an RSU receives a registration message from a regular vehicle, it checks the message is new to send it directly to the TA and waits for the confirmation of registration from the TA. The RSU saves the vehicle information when it forwards the message to the TA. Whenever an RSU receives a registration confirmation from the TA, it checks if it is a new message to store it and then announces it multiple times. An RSU can announce the message whenever a beacon message from the same vehicle is received. The RSU can optionally send this message to nearby RSUs. Also, as the TA maintains the neighbour list of RSUs, the TA can send the confirmation to all neighbouring RSUs to forward the registration message. In this way, the vehicle and/or the driver can receive the registration confirmation message quickly. Fig. 3-31 shows a flowchart for registration message management at an RSU.

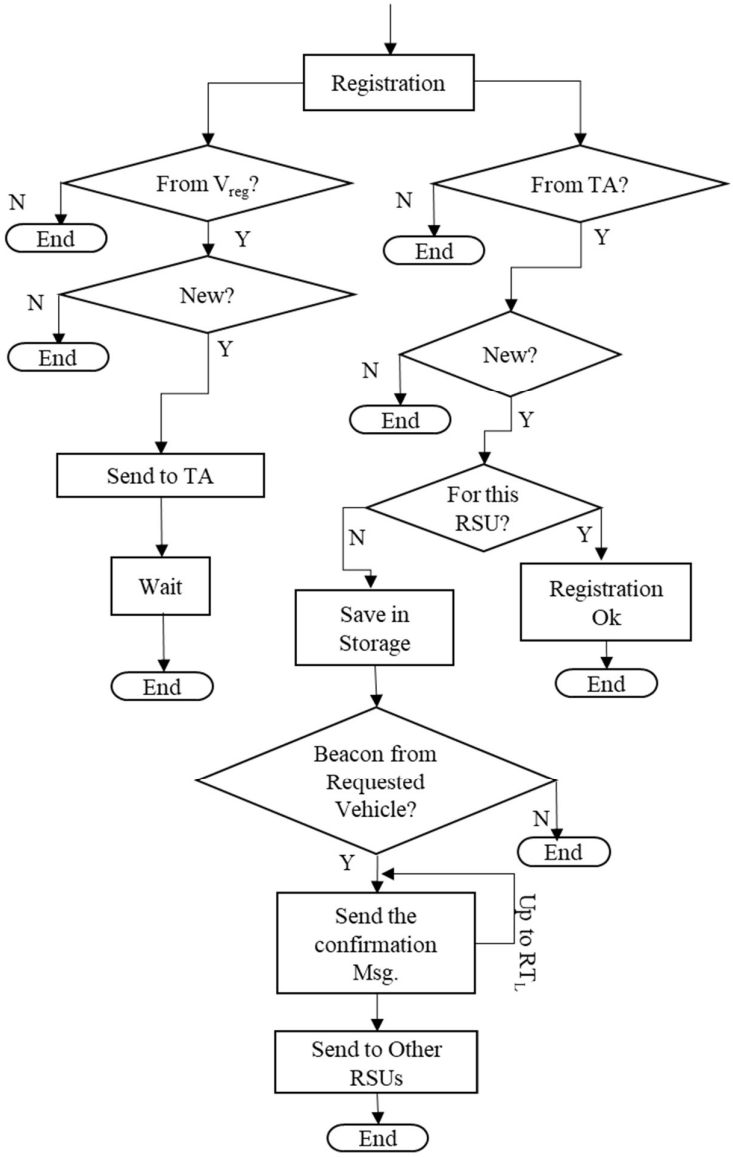


Fig. 3-31: Flowchart for Registration Message Management at an RSU

When an access-blocking message is received by an RSU, it checks if it is from a regular vehicle or another RSU, or the TA. If the source is a regular vehicle, then the RSU checks whether it is a new message to store it and then forwards it to the TA and optionally sends it to nearby RSUs. When an RSU receives an access-blocking message from another RSU, it saves it into storage and checks if it is a blocking confirmation message. If this is the case, it announces the message multiple times, otherwise, it simply saves it. If there is an access-blocking message from the TA, then the RSU checks if it is received first time to store it. If new, it is announced up to configurable times, and it is also optionally relayed to nearby RSUs. Also, as the TA has the neighbour list of RSUs for each RSU, the TA can send the confirmation to adjacent RSUs. In this way, the vehicle and/or the driver can receive the access-

blocking confirmation message more readily. Fig. 3-32 shows a flowchart for access-blocking message management at an RSU.

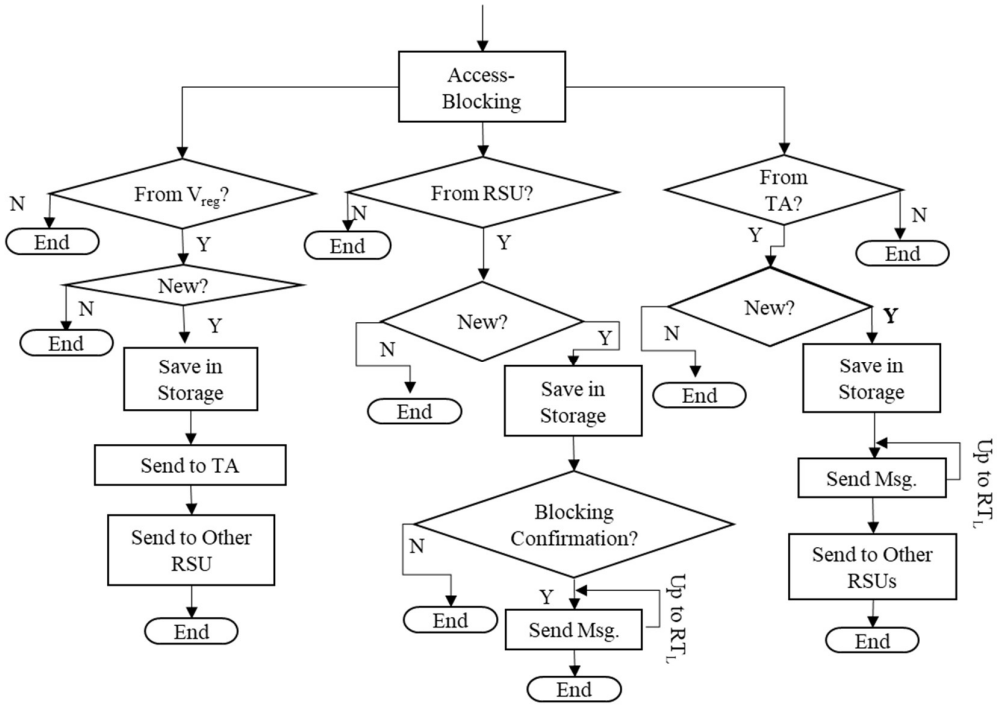


Fig. 3-32: Flowchart for Access-Blocking Message Management at an RSU

3.11.4 Trust Authority

We assume the TA is first initialized with its resources at the INIT step. The TA is mainly responsible for dealing with registration, access-blocking, decision of untrue situations, updating events into incident database, and storing the trust scores of drivers as well as dealing with road-issues. Fig. 3-33 shows the primary roles of the TA.

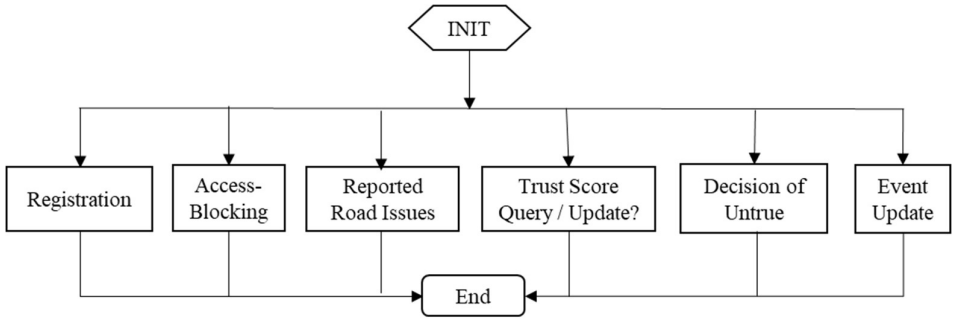


Fig. 3-33: Flowchart of Main Activities Performed by the TA

For a reported road issue, the TA checks whether the message is new. If the message is new, then it checks it is one of the messages that the TA handles. If this is the case, it stores the message in storage and then contacts the RTA for every road issue to allocate personnel and resources to deal with the road

issue. During this period, it waits until there is a notification from the RTA about the resolved issue. The TA then prepares a resolved road-issue message based on the RTA notification and sends the message to the sender RSU. Fig. 3-34 shows a flowchart for handling road issues by the TA with the RTA.

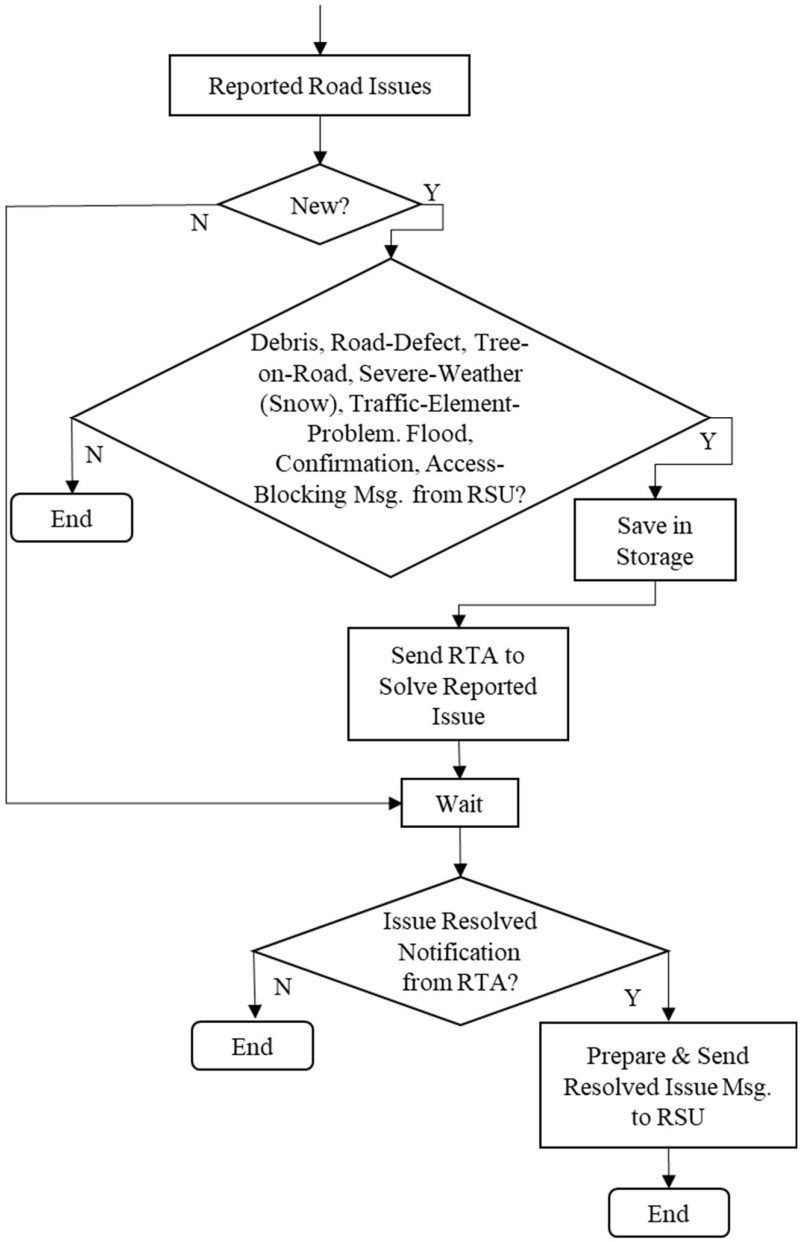


Fig. 3-34: A Flowchart for Handling Road Issues by the TA Through RTA

Whenever a registration request is received either from a vehicle or from an RSU, the TA follows the same set of actions. It first checks if it is a new request to start the registration. Then it checks whether the vehicle or RSU is already registered. If it is not, then it stores the request and adds the vehicle and/or driver or RSU information to a confirmation list. After this, it prepares and sends a

confirmation message to neighbouring RSUs of the sender RSU if it is for vehicle registration. In case of an RSU registration, it directly sends it to the sender RSU. Whenever a decision of an untrue attack arrives at the TA, it checks whether it is received for the first time to store the decision. Additionally, it checks whether three malicious counts have arisen for this driver to add the driver to the access-blocked list and issues an access-blocking confirmation message to the neighbouring RSUs of the RSU from which it received the most recent decision. Fig. 3-35 shows a flowchart for registration, access-blocking of vehicles and/or drivers, untrue decisions, and trust score management at the TA.

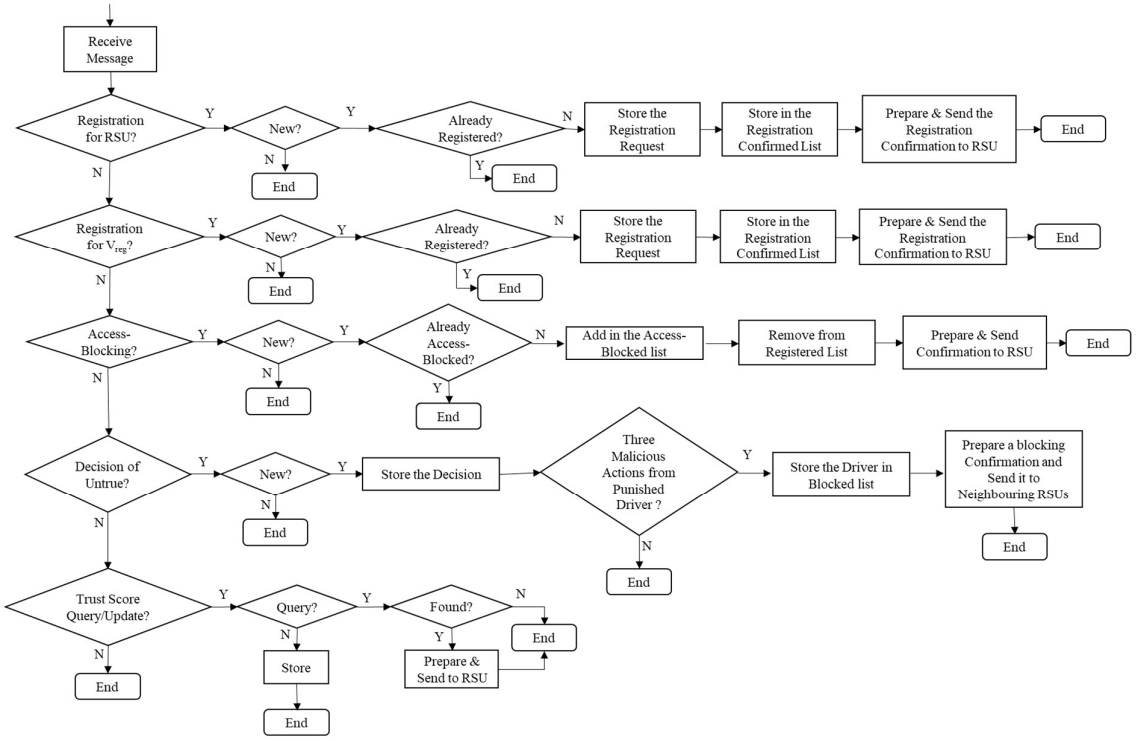


Fig. 3-35: A Flowchart for Registration, Blocking, Untrue Decision, and Trust Score at the TA

3.12 Cooperation Attack Detection with the Proposed Framework

Suppose two known people are driving towards a destination. While they are driving, they get closer to each other, but later, one driver falls behind another driver due to any reason. The rear person may request the front person using a cellular network or other means to send a traffic jam / congestion message that is an abuse of the VANET. With this message, the vehicles in front of the rear driver may be discouraged from using a part of the route ahead of them. This is an illustration of a cooperation attack. However, with this framework, whenever a person sends a false traffic jam / congestion message, other vehicles immediately following him / her treat it as an untrue message and send a report to a nearby RSU. Using RSU collaboration, the driver who generated this untrue event will be punished by the framework. In this way, this form of cooperation attack can be thwarted. This scenario is depicted in Fig. 3-36.

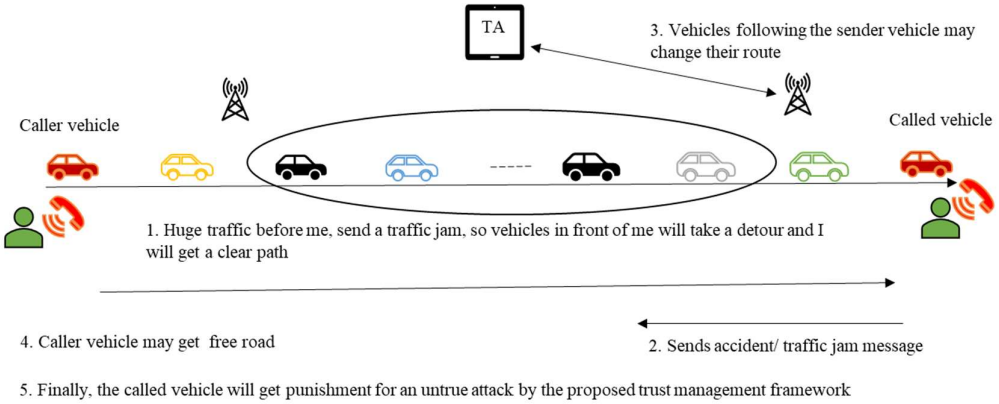


Fig. 3-36: Scenario of Cooperation Attack and Punishing a Vehicle

3.13 Summary

This chapter presents the proposed trust management framework. It first presents the elements of the framework and the trust evaluation mechanism for regular vehicles. In this framework, trust is the primary means of access control for message announcements and relaying from regular vehicles. Then an algorithm is presented to show how traffic events are managed and processed by regular vehicles. There is also an algorithm for updating trust and access-blocking management at the TPD. Besides this, a functional diagram of the proposed framework is also given. As any trusted vehicle and/or driver can announce untrue messages in the VANET, these are thwarted through a collaboration process running at the RSU. This framework can also detect inconsistent attacks using this process. For this, an untrue detection algorithm is provided, and another algorithm issues the reward and punishment after deciding upon the validity of a dispute. After this, interactions among the various entities are shown in flowcharts.

In the next chapter, simulation, modelling, validation, and a performance evaluation of the proposed trust management framework are considered.

Chapter 4: Implementation, Validation, and Performance Evaluation of the Proposed Trust Framework

4.1 Introduction

In the previous chapter, the proposed trust management framework is presented in detail considering various traffic scenarios. In this chapter, first, the simulation and modelling of the proposed framework is highlighted and then the validation and performance evaluation of the framework is addressed. To implement this framework a suitable simulator that can visualize real-world traffic mobility is required. Many network and traffic simulators exist for simulating a VANET. This chapter starts with a brief discussion of existing simulators and then highlights the Veins simulator. Furthermore, this chapter also describes the step-by-step upgrade to the Veins simulator to enable it to model the capabilities of the proposed framework. Various scenarios for different traffic events are then simulated including the message interaction. During the validation and performance evaluation of the proposed model, untrue and inconsistent attacks (on-off) are disseminated, identified, and the appropriate entities are punished to limit their future malicious actions (thwarting). To this end, a set of experiments is carried out to demonstrate the capability of the proposed trust model in thwarting these attacks. Additionally, one form of cooperation attack can be thwarted which is theoretically analysed in Chapter 3. After this, the accuracy of the proposed trust framework in response to malicious and benevolent feedback from clarifiers is considered. Finally, the framework is compared against a well-known reputation model in terms of communication overhead and response time.

4.2 Network and Traffic Simulators

In vehicular network simulations, both network and traffic simulators can be used together. A network simulator can measure the efficiency of network communication. In contrast, a traffic simulator visualizes traffic mobility along with message communication. These two simulators are coupled together when they require online communication to visualize the communication. This section first surveys network simulators. Then traffic simulators are discussed briefly. After this, some combined network and traffic simulators are considered. From this review, Veins is selected as it is available freely and is a combined simulator. Additionally, a great deal of VANET research has been analysed using Veins.

4.2.1 Network Simulators

It is not always feasible to examine a communication protocol using only real-world networking devices, especially vehicles as they are costly. This is why researchers often use an alternative cost-

effective solution which is running experiments in a simulation environment where vehicles can be modelled to represent them in the real world. Additionally, simulators can achieve results that are readily acceptable to academia. To this end, many network simulators have been developed including ns-2 (https://www.isi.edu/nsnam/ns/index.html [122]), ns-3 (Zhang et al [123]), OPNET (Chang et al [131]), OMNeT++ (Varga et al [18]), QualNet (Quality Network) (Dinesh et al [159]), and NETSIM (Rathi et al [160]).

The ns-2 simulator is an open-source, discrete event simulator used by network communication researchers for simulating TCP/IP protocol, multi-hop communication, and multicast routing protocol both for wired and wireless mediums. It is written in C++ and users can interact with it using Objective Tool Command Language (OTcl) and view output in NAM (Network Animation) editor. One of the shortcomings of the ns-2 is to set up a manual connection from each node to all its neighbours. The ns-3 simulator (www.nsnam.org [154]) is an extension of ns-2 which supports both simulation and emulation features. ns-3 has an additional interface for running Python script and open-source software can be integrated through built-in interfaces.

OPNET (Chang et al [131]) is a commercial discrete event network simulator. It is used to analyse the behaviour and performance of both wired and wireless networks. It includes pre-built protocols and devices. QualNet (Dinesh et al [159]) is also a commercial, scalable, and fast simulator designed for both wired and wireless network simulation. QualNet can be configured and used with most operating systems. QualNet is used to evaluate the new protocol modelling, simulation, and performance analysis for Wi-Fi, WiMAX, MANET, and sensor networks. NETSIM (Rathi et al [160]) is an alternative commercial network simulator used for both wired and wireless network simulation and analysis. NETSIM (Rathi et al [160]) can be used for TCP/IP, GSM, LTE, IOT, MANET, and VANET network simulation and performance analysis.

OMNeT ++ (Varga et al [18]) is a discrete event, freely available network simulator that is widely used in academia. This simulator is written in C++ and can simulate and analyse the performance and behaviour of TCP/IP, ad hoc networks, sensor networks, and VANETs. Its design is based on modular architecture and hence new features and components can be added in the form of simple and complex modules. In OMNeT++ a network is treated as a complex network module built upon a hierarchy of simple modules. Many extensions and frameworks are also available to work with OMNeT++ for example, MIXIM suite, INET, and the Veins framework. In OMNeT++ wireless network scenarios are created from simple and compound modules, and the network module is specified in a (network descriptor) NED file. For every simple module in OMNeT++, there is a separate C++ implementation file to define the functionalities of the simple module and a header file to declare all the required resources. These three files (NED, CPP, and header) share the same name as the simple module one.

Simple modules are used as submodules and placed inside of a compound module or network module using the NED editor. There is no need to include the implementation of complex modules or network modules in OMNeT++. Simple modules connect with the compound module using gates and connections. Compound modules can also be connected to other compound modules. For each simple module, gates, and connections are created first and then they are connected using the connections in the NED language. In OMNeT++ input, output, and bidirectional gates are available. To simulate a network in OMNeT++ one must specify all the network parameters in a separate configuration file called omnetpp.ini. In this file, we specify the simulation scenario including the total simulation time the simulator runs for data collection, nodes that send messages including their time of message generation. Some of the parameters can also be set from the NED file. To send messages in OMNeT++ one needs the cPacket and cMessage classes.

4.2.2 Traffic Simulators

Simulation of Urban Mobility or SUMO (Behrisch et al [161]) is an open-source tool that supports both microscopic and macroscopic traffic classes. It is a GUI-based continuous traffic simulator that is designed for large-scale traffic flow simulations. In SUMO individual vehicle properties can be set. SUMO supports multiple lanes, lane changing, speed variation, different vehicle types, a car-following mobility model, intersections, and traffic lights. SUMO supports importing different network formats, for example, VISUM, VISSIM, shapefiles, OPENSTREET MAP, and so forth. SUMO was initially designed for traffic planning and road design. It is now also used in traffic engineering research. The NETEDIT of SUMO (Behrisch et al [161]) enables the creation of road networks by defining the multiple lanes where vehicles can move at varying speeds. During the road creation, a vehicle's insertion point, and exit point are selected. Mobility Model Generator for Vehicular Networks or MOVE (Lan et al [162]) is a JAVA-based tool that extends SUMO. MOVE (Lan et al [162]) has a feature to import maps from Google Earth and TIGER databases. There is a map editor and visualization tool in MOVE (Lan et al [162]) to view a mobility trace. VanetMobiSim (Härri et al [163]) is an extension of the CANU Mobility Simulation Environment (CanuMobiSim) which is an open-source traffic simulator, written in JAVA. This simulator supports mobility traces in different formats and can be used with several network simulators, for example, ns-2, QualNet (Dinesh et al [159]), and GlomoSim (Zeng et al [164]). VanetMobiSim (Härri et al [163]) focuses on vehicular mobility and includes both macroscopic and microscopic motion models. This supports multi-lane roads, separate directional speeds, and traffic signs at the intersection with the macroscopic model. Alternatively, the microscopic model allows V2V and V2I interaction. Vehicles can move by maintaining distance from other vehicles, can overtake each other, and can follow signs at intersections.

4.2.3 Combined Simulators

We need to select an effective tightly coupled simulator to test the behaviour and performance of the proposed trust model in a VANET setting. The network simulator only evaluates the networking performance issues. Conversely, a traffic simulator visualizes the mobility of vehicles considering various traffic scenarios on roads. There are several tightly coupled network and traffic simulators which are discussed next briefly. Then one of the combined simulators is selected for this research, which is the Veins simulator.

Veins (Sommer et al [17]) is an open-source, tightly coupled framework for simulating vehicular networks comprising traffic simulator SUMO and discrete event simulator OMNeT++ (Varga et al [18]). In Veins, SUMO provides traffic flow and a map of a road network that can be imported using OpenStreetMap (Bennett [165]) or can be designed using NETEDIT of SUMO. NETEDIT is built-in software that comes with SUMO to create road networks and insert vehicles by setting properties. OMNeT++ provides networking protocol simulation including the application layer, DSRC, and physical layer to emulate realistic network communication. Communication between SUMO and OMNeT++ is achieved online through Traffic Control Interface or TRaCI (Wegener et al [166]). Whatever happens in OMNeT++ (Varga et al [18]), TRaCI sends the commands to the vehicles in SUMO and vice versa.

In SUMO, a user can design road networks using the NETEDIT and then he/she can specify the routes where the vehicles run during the simulation with the departure and arrival times. Veins can add single or multiple traffic flows that are created using SUMO. When defining the traffic flows in NETEDIT, the vehicle insertion point, period of insertion, and routes are specified. Traffic flows and vehicle information are saved in the .rou.xml file for the network created in the .net.xml file with the same name. After that, the vehicles that send messages, and their time of message generation are set from the omnetpp.ini file. The config file omnetpp.ini in Veins specifies this information about the path loss model, obstacles, the decider model, and the centre frequency at which the physical layer listens. Parameters can also be specified in the NED file. The architecture of Veins has only three layers which are the application layer, the mac layer, and the physical layer. Also, the Mac layer and physical layer are merged into the NIC layer. Each vehicle has a separate mobility submodule to advance the vehicle at every timestep.

One can develop a simulation using a graphical user interface (GUI) based editor and run programs both in GUI and command mode in Veins. OMNeT++ provides QtEnvir and TkEnvir graphical environments to run programs in GUI mode. Users can watch and inspect the scenarios running on the GUI while a user can interact with the GUI environment with a stepwise or express mode or normal running mode. Also, OMNeT++ and Veins support batch mode execution of the same simulation using

random seeds so that multiple sets of outputs can be generated together from the same scenario. Furthermore, users can see the events in an event log.

It is possible to generate scalar and vector data from OMNeT++ based Veins to analyse the performance of any model. Users can select a configuration from a list of configurations that are defined in the omnetpp.ini file. Users can also see how an event calls other events in the sequence via a SEQUENCE diagram that the OMNeT++ output environment generates. In OMNeT++, all created events are inserted into a cEventHeap from which the scheduler fetches one event to process at a time according to the priority of the event. There is a separate xml launched file where TRaCI finds this information to connect with the SUMO which acts as the TRaCI server and OMNeT++ works as the TRaCI client in Veins.

Traffic and Network Simulator environment or TraNS (Piorkowski et al [167]) is designed solely for VANET simulation which allows changing the behaviour of vehicles in a mobility model. In TraNS (Piorkowski et al [167]), projects can be written in both JAVA and C++. It features the Google Earth visualization and TRaCI-based integration of SUMO and ns-2. TraNS (Piorkowski et al [167]) has two different architectures which are network-centric and application-centric modes. In the network-centric mode, one can evaluate vehicle mobility and communications without considering real-time vehicle movement. Whereas, in the application-centric mode, one can evaluate the VANET application considering real-time movement. This simulator feeds the output from SUMO into ns-2, but the reverse is not possible.

National Chiao Tung University Network Simulator or NCTUns (Wang et al [168]) is also a tightly coupled GUI-based traffic and network simulator and emulator, written in C++. It can perform simulations for both wired and wireless networks. For example, users can simulate ad hoc networks, VANET, WIMAX, LAN, GPRS cellular networks, optical networks, and mesh networks. It can simulate most of the MAC protocols and Internet protocols. Protocol developers can easily integrate their protocols within the simulation engine. Additionally, it can execute jobs from a remote request and when the job is done it sends the result back to the client to display in the GUI. However, it is now only commercially available and is renamed EstiNet.

Among these combined traffic and network simulators, a tightly coupled simulator is a preferable option for simulating a VANET as it is convenient to observe the real-time interaction among entities between the network and traffic simulator. Considering this, NCTUns and Veins are the two most prominent options. However, as NCTUns is now only marketed commercially as EstiNet, Veins was elected for simulating the proposed trust model as it is an open-source framework and offers unrestricted extensibility. It is based on IEEE 802.11p and IEEE 1609.4 DSRC/WAVE network layer, QoS channel access, noise, and interference effects (Sommer et al [17]). It can import road traffic scenarios using OpenStreetMap (Bennett [165]) including buildings, speed limits, lane counts, traffic lights, access and

turn restrictions. It combines SUMO and OMNeT++ into a single framework. The functionalities of the lower layers are already implemented including message transmission, mobility management, simulation set-up, execution, and results collection during and after the simulation. Thus, Veins is an appropriate option to simulate this trust model. In addition, many researchers have also used Veins to simulate many trust management frameworks (Gazdar et al [12], Kumar et al [33], Chandra et al [40], and Al-Riyami et al [47]).

4.3 Implementation

The framework is implemented in Veins 5.0. The simulation model of the proposed approach is evaluated on a computer with 8 gigabytes of RAM and an Intel core i5 quad-core processor with each core running at 1.6GHz. To evaluate the proposed trust model, we need a scenario where most of the elements of a VANET are present. The primary elements are vehicles, RSU, and TA. We consider regular vehicles to be the primary users. To facilitate communication, authority monitoring, and event management, we need official vehicles, roadside infrastructure and one trusted entity who can officially register and access block vehicles. Official vehicles are included as they are common and serve a specific purpose on the road. For example, a police vehicle inspects whether vehicles are obeying rules while driving, an ambulance gives primary treatment on the go, and they carry the injured and ill patients to the hospital. These are the most common classes of vehicles and are considered for the analysis. However, there exist many classes of vehicle, for example, taxi, bus etc., which we do not consider at the moment though they could be added easily in the future. Additionally, official vehicles are an integral part of this framework as their feedback is authentic when resolving a dispute by the RSU. We also need infrastructure facilities along the road to collect data from the vehicles and to facilitate timely information dissemination among the users and nearby regions. Hence, RSU inclusion within the scenario is required. Furthermore, network incident information, driver history of reward / punishment should be stored at a trusted site besides supporting registration and access blocking for vehicle / driver. The TA performs these activities in this model. Thus, we have included regular vehicles, official vehicles, RSUs, and a TA in the simulated scenario. To incorporate these entities, Veins is extended in several respects. First, four types of vehicles are created including “official” vehicles (police, ambulance, and fire service) and regular vehicles. Aside from these modules, the TA does not need to have wireless broadcasting capability, and the RSUs remain stationary during the simulation. The TA module primarily registers and blocks drivers. Additionally, the TA unit keeps each driver’s most recent reward/punishment history in a driver profile database to facilitate the blocking of malicious drivers and record incident information (location, timestamp, incident) in an incident database. In addition to this, an RSU internetwork is developed which also connects to the TA unit via wired connections. Inside each RSU, besides the event management, the dispute resolution process is implemented to detect untrue/inconsistent attacks. There is a built-in mobility module from Veins that advances all vehicles at regular time steps. This way Veins provides a realistic environment for the simulation of a VANET.

Additionally, different messages are created to simulate some real-world scenarios and we assume vehicles generate appropriate messages when they notice an event on a road.

The vehicle application first reads the trust from the TPD. This trust needs to satisfy the associated trust threshold for the message class to proceed with the announcement. When a sender sends a message, three metrics are collected which are the truthfulness of the event (the message is reliable if there is no report at the sending entity), delay / responsiveness (driver inserts event notice time from a timer and announcement time from pressing a button), and distance travelled after noticing the event from each announcement. When drivers broadcast messages, the application sends the delay and the location of the event from the current location of the vehicle identified on the map to the TPD. These metrics are used by the TPD to compute the reward/punishment for the announcement using Algorithm 3-2. A TPD module is added to regular vehicles which primarily implements the trust update and access-blocking logic of each driver. This module exchangess messages with the vehicle application layer using an internal connection. The TPD withholds rewards for a given period in case a dispute arises via reporters. If there is a report within this time, the TPD does not add the reward. After this, the TPD expects a decision from nearby RSU that resolves the dispute. If there is no report, the TPD adds the reward for the announcement. For other activities, the TPD adds the reward or deducts the punishment, as necessary. Also, the TPD disables the transmission of a blocked driver to stop the generation of event announcements; however, whilst in the blocking state a driver still broadcasts beacons. The reward varies for activities like beaconing, forwarding, and broadcasting announcements. Vehicles can only obtain a beaconing reward if they are classified as *not trusted* or *lowly trusted* as defined in Chapter 3. The TPD can also support multiple driver profiles in case different people share a vehicle. RSUs send rewards/punishments to the respective entity based on their decision.

4.3.1 System Model and Environment

When Veins (OMNeT++ and SUMO) was initially installed, tests were carried out to observe how the messages are generated, and how the vehicles react to the messages. For this evaluation, only one RSU and many vehicles of one type were running on the Erlangen city map. Later, multiple types of vehicles and more RSUs were added to the map, as needed. This involved the RSU interconnection network and their connection with the TA. The creation of new maps was explored with Veins using, for example, using OpenStreetMap or NETEDIT of SUMO. However, the Erlangen map is convenient rather importing a new one for conducting simulations as it shows the buildings and roads together on the map. A Manhattan grid and two alternative route scenarios were created using NETEDIT of SUMO. In these maps, multiple routes were configured using NETEDIT and multiple traffic flows with each vehicle's insertion point on the route. Vehicles are added one after another from this point during the simulation. The traffic density can be controlled by changing the periodicity of vehicle insertion. Veins by default employs the SUMO car movement model. The traffic flows were verified in SUMO to check

that all the vehicles can make their journey successfully from the beginning to the end. When the execution of a traffic scenario is initiated, TRaCI fetches this information from a launched file and creates a node in OMNeT++ for every vehicle created in SUMO. In OMNeT++, the state of vehicles (position, speed, acceleration) is updated according to the vehicle state data from SUMO which is updated at each timestep. The vehicle's communication is governed by the vehicle application layer in Veins.

As there are four types of vehicles, four distinct modules are created in OMNeT++ and the C++ implementation of them according to the functionality specified in Chapter 3. Additionally, different application layers are created for the TPD, RSU, and TA. For each application layer, three files are defined which are a NED file, an implementation file, and a header file. In the NED files, parameters, gates, signals, and statistics are defined for the corresponding module. In the implementation file, the functionalities of the application layer are written in C++. All the declarations of all variables, data structures, and methods that are used in the implementation file are identified in the header file. These applications run as submodules of entity modules, for example, regular vehicles, police, RSU, and TA. These entity modules consist of one application layer submodule, one network interface card (NIC) submodule, and one optional mobility submodule. There is also a network module in OMNeT++ which is added to all entity modules to create the VANET. This network file is extended by RSUExampleScenario in Veins which declares the number of RSUs, the TA and their interconnections. Messages originate from some selected vehicles which are relayed by other vehicles to reach RSU. However, their relaying is limited by a configurable hop limit. In this way, controlled broadcasting through flooding is achieved. In OMNeT++, messages are created by extending cMessage / cPacket class and adding suitable fields.

The application layer of official vehicles is different from regular vehicles. Official vehicles respond to RSU queries differently than regular vehicles. Also, when an emergency arises, an RSU gives precedence to their messages over regular vehicles. The system starts with no vehicles in the terrain model and once a vehicle is added, it remains in the system until the simulation terminates. Once a predetermined number of vehicles enter the system, no more are permitted. The simulation commences by assigning periodic events to specific vehicles and then the resultant data are collected regarding the specific experiment. The framework is simulated using the Erlangen city map (Sommer et al [17]) as shown in Fig. 4-1a, the Manhattan grid map in Fig. 4-1b, and one alternate route scenario in Fig. 4-1c.

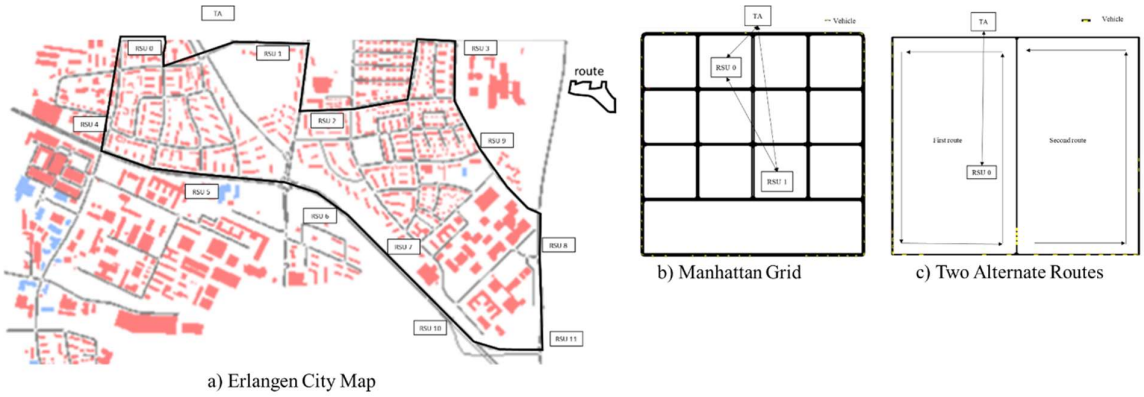


Fig. 4-1: Road Networks Used in the Simulation

4.3.2 Implementation of Communication Scenarios in Veins

Within the various scenarios the message types shown in Table 4-1 were created for the analysis of the proposed trust model. Regular vehicles can send a registration message when they start their journey if the VANET supports online registration. RSUs can also be registered with the TA with a registration confirmation message. In this way, the TA is informed about the vehicles and RSUs that are using the VANET. To create an accident scenario in Veins, first an "Accident Message" is created by extending the cMessage / cPacket class of OMNeT++ and adding all the necessary fields to the message, for example, road information, driver and/or originating vehicle, vehicle position on road, observation and sending time and so forth. The application layer of the vehicle creates this message and then adds other associated information to forward it to the lower layer to broadcast by the physical layer into the wireless environment. Also, the sender vehicle schedules a timer to retransmit the same message several times which can be configurable. When receiver vehicles receive this message, they check the message and, if appropriate, possibly amend their route, and forward the message to more distant vehicles if the hop limit is not reached. Whether a vehicle will detour or not is decided by the driver. When this message reaches an RSU, then it repeatedly warns the neighbourhood until the issue is resolved. Meanwhile, if an official vehicle is nearby, it sends a roadClearMessage (event-sorted) update to a nearby RSU when the issue is resolved. If any RSU receives a roadClearMessage, then this RSU informs nearby RSUs. After this, they broadcast the event-sorted message a number of times so that the vehicles that are possibly affected do not need to detour. Other traffic situations can be handled in a similar manner with the proposed trust model.

Table 4-1: Messages Considered with the Proposed Trust Framework

Beacon Message	DemoSafetyMessage
WSA message	DemoServiceAdvertisement
Wave Safety Message (WSM)	Accident Message, Traffic Jam, Traffic Congestion, Diversion, Road Maintenance / Road Closure, Speed Limit, Stranded Vehicle, Obstacle-on-Road, Obstacle-Clear-on-Road, Foggy / Ice / Mud / Stones-on-Road,

	Poor-Conditioned-Road, Regular Visit, Debris On Road, Flooding On Road, Tree On Road, Road Defect, Traffic Signal Problem, Parking Locator, Restaurant Locator, Petrol Pump Locator, Wi-Fi Point Locator.

4.3.2.1 Verification of the Proposed Framework

4.3.2.1.1 Vehicle and RSU Registration

If the authority wants vehicles to register, then they register themselves by sending a "registration request" message to the TA via a nearby RSU through the RSU's direct interconnection with the TA. The TA confirms the registration with a message saying "registration done for vehicle V or driver D" to some neighbouring RSUs of the RSU that forward the message. The RSUs broadcast the registration confirmation message to reach the appropriate vehicle. An RSU can also register with the TA if it has not already done so. A sequence diagram for the registration process for both an RSU and a vehicle is shown in Fig. 4-2.

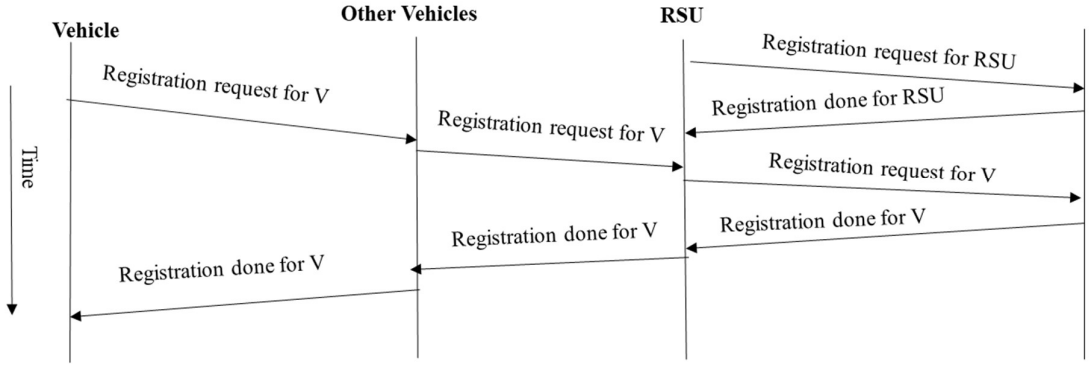


Fig. 4-2: Illustration of Registration of Vehicle and RSU in the Proposed Framework

In Fig. 4-3, initially, vehicles A, B, C, W, X, Y, and Z broadcast registration request messages to their nearby RSU. This RSU forwards these messages to the TA. The TA first checks whether the vehicles and/or drivers are already enrolled within the VANET. If a vehicle is not registered then it enrols the vehicle and saves its identity information into its confirmed registration list. After this, the TA replies to each vehicle's registration message separately with a confirmation message via a set of RSUs. The confirmation message eventually reaches the vehicle concerned. After this, they can receive services or broadcast messages based on their trust score in the VANET.

Initially, RSUs directly send a "registration request" message to the TA. The TA stores the identity of RSU in its registered list. The TA generates a "registration done for RSU" message to the corresponding RSU. In this way, RSUs are registered with the TA. Official vehicles including police, ambulance, and fire engine trucks are not required to perform registration in this way as they already belong to the authority. Fig. 4-3 illustrates the registration process for regular vehicles.

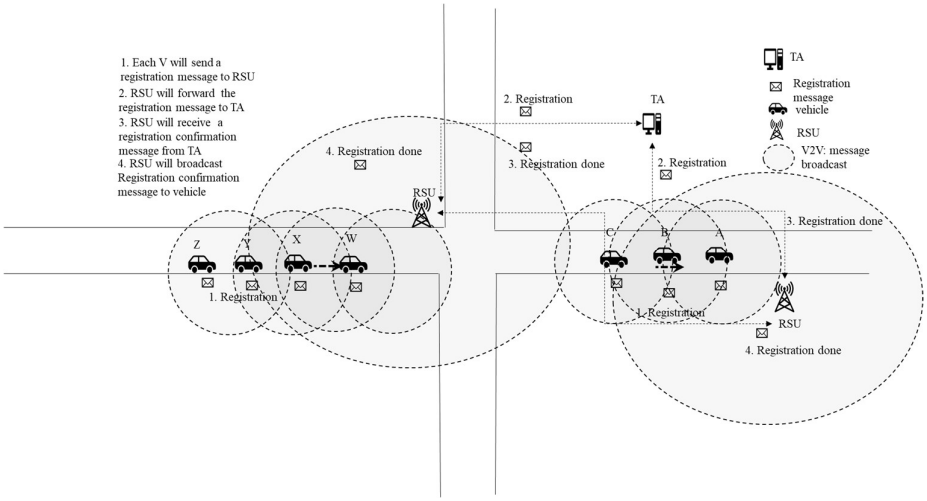


Fig. 4-3: Broadcasting of Registration and Confirmation of Registration Message

4.3.2.1.2 Access-Blocking of a Vehicle

If the trust score of a driver / vehicle becomes 0.05 after several malicious activities, then the TPD of the vehicle initiates an "access-blocking" process for the driver/vehicle. This message is broadcasted by the application layer including the trust T<=0.05 towards a nearby RSU. This message may be forwarded through some intermediate vehicles. The RSU forwards the access-blocking message to the TA. The TA inserts the driver / vehicle information in the access blocking list. Then the TA sends a "confirmation of access-blocking" message to a set of neighbouring RSUs of the RSU that forwarded the original message. These RSUs broadcast the confirmation message several times to reach the recently blocked vehicle. Finally, when the vehicle receives this message, the TPD disables the transmission capabilities of the concerned driver / vehicle. Transmissions from the driver / vehicle will not be permitted until an external redemption is followed, a monetary penalty is paid, or whatever the RTA decides. Fig. 4-4 depicts the sequence diagram for access-blocking. In Fig. 4-5, vehicle Y broadcasts an access-blocking message with its trust score of 0.05 to reach the TA via an RSU. Also, the sequence of message exchanges is shown. A driver / vehicle is also access-blocked by the TA if the count of malicious activity reaches a configurable limit (for example three).

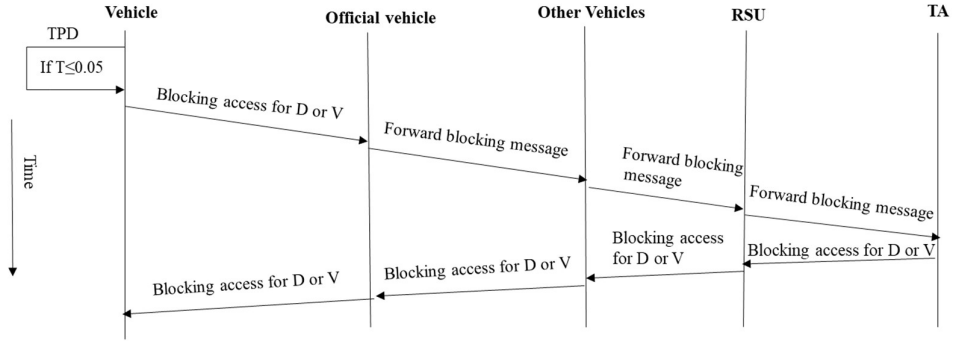


Fig. 4-4: Sequence Diagram for Access-Blocking

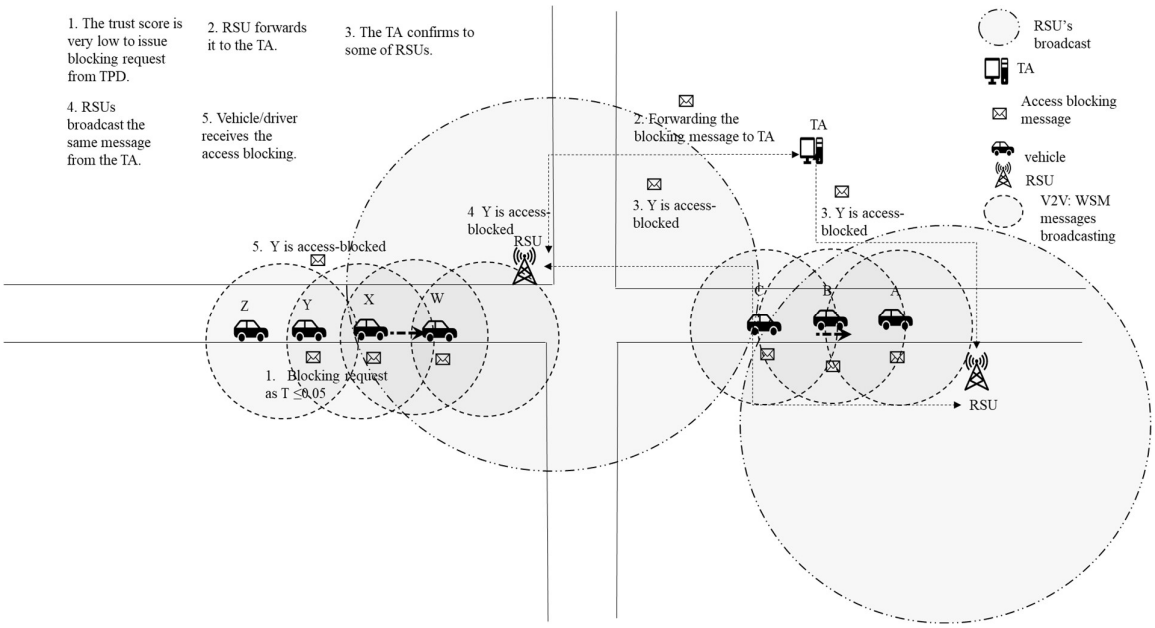


Fig. 4-5: Access-Blocking of a Driver / Vehicle

4.3.2.1.3 Periodic Beacons

Vehicles A, B, C, W, X, Y, and Z in Fig. 4-6 broadcast beacon messages periodically which are received by others whenever they are in direct transmission range. In the same way, each RSU also broadcasts beacon messages periodically in the VANET. If a vehicle receives a beacon message from an RSU it means they are in direct transmission range of each other.

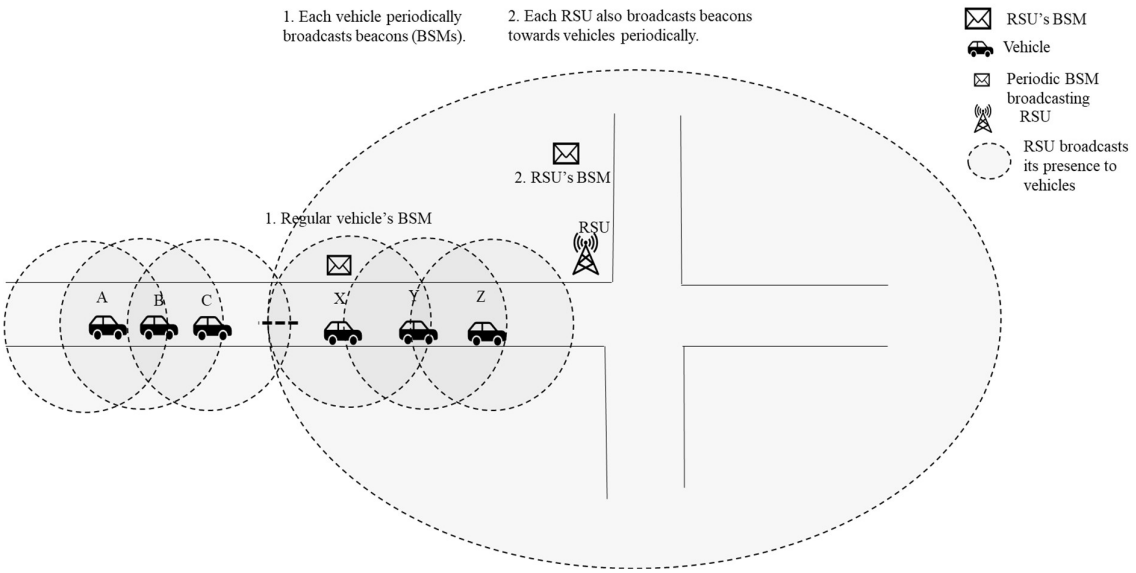


Fig. 4-6: Broadcasting Beacon Messages

4.3.2.1.4 Announcement of an Accident

Suppose an accident occurs on a road named “X” and a vehicle notices the crash while it is driving along this road. This vehicle broadcasts an accident message to other vehicles and RSUs. When an RSU receives this message from intermediate vehicles through relaying, it also retransmits the same message in the VANET. Also, the RSU periodically broadcasts an “avoid road” message since road “X” is unavailable. The RSU also communicates with nearby RSUs to broadcast the “accident on road X” message so that probable traffic chaos can be avoided near the event. Vehicles that obtain the announcement from any source may avoid road “X”. Meanwhile, if a police car receives the announcement, then it broadcasts an “attending road X” and a “free road X” message multiple times to reach the incident location promptly. If an RSU receives this message from a police car, then it announces a “restricted movement on road X” message. Whenever an RSU receives a message from a police car, the RSU periodically broadcasts a message saying, “On road X, only police, ambulance, or fire service vehicle will enter with the highest priority, and other vehicles should use a different route”. This creates awareness among the nearby drivers so that all the regular vehicles may avoid road X. Hence, official vehicles can reach the incident location early. As a result, the incident can be resolved faster. When the incident is addressed, the police vehicle sends a “sorted road X”. Finally, the RSU broadcasts this sorted-road message multiple times periodically so that vehicles can use road X again. The RSU also sends the sorted-road message to nearby RSUs to broadcast the same message multiple times. In this way, an accident event is addressed with the proposed trust management framework. Each message for example “accident message on road X”, “avoid road X” is declared as a packet that extends Veins default packet named “BaseFrame1609_4”. These messages contain the necessary fields and meaningful information. For example, “attending road X” signifies an official vehicle is approaching an event to investigate. In contrast, “sorted road X” means that the said event is now cleared on road X. Each message name suggests the event type to its users. This applies to all network event messages created for evaluation of the proposed trust framework. Fig. 4-7 depicts a sequence diagram announcing an accident message. Fig. 4-8 illustrates the overall process of accident event handling, and Fig. 4-9 shows the resolution process associated with an accident.

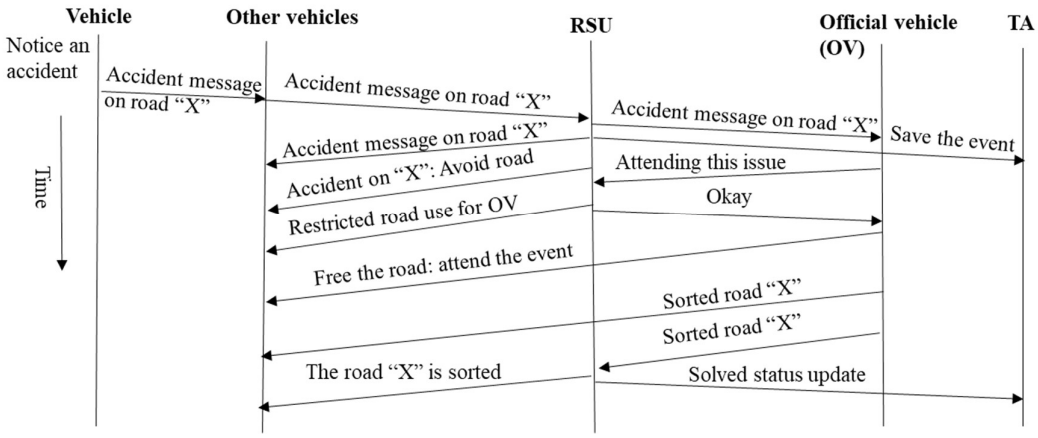


Fig. 4-7: Sequence Diagram of an Accident Event Announcement

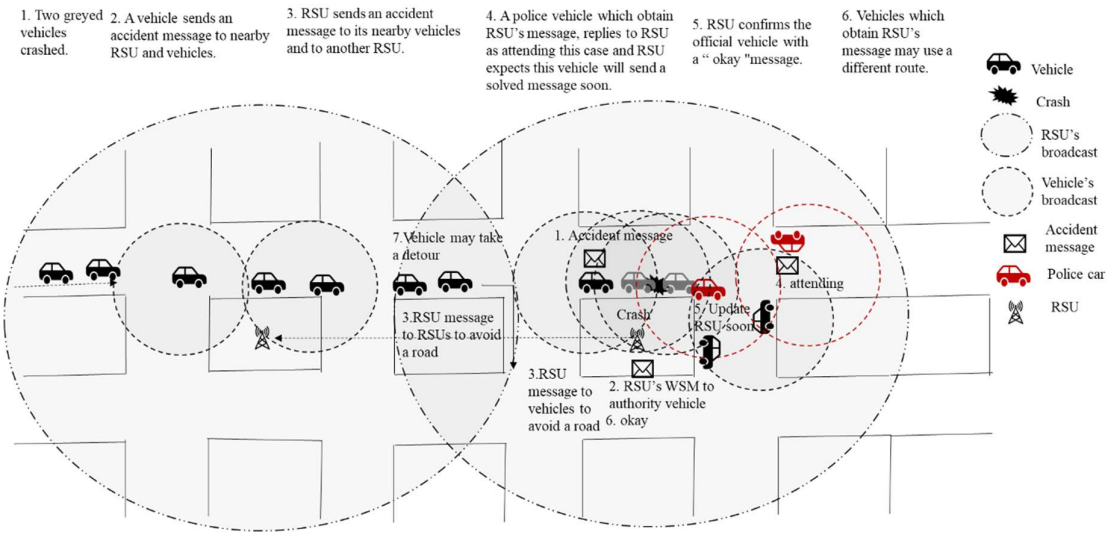


Fig. 4-8: Accident Message Announcement

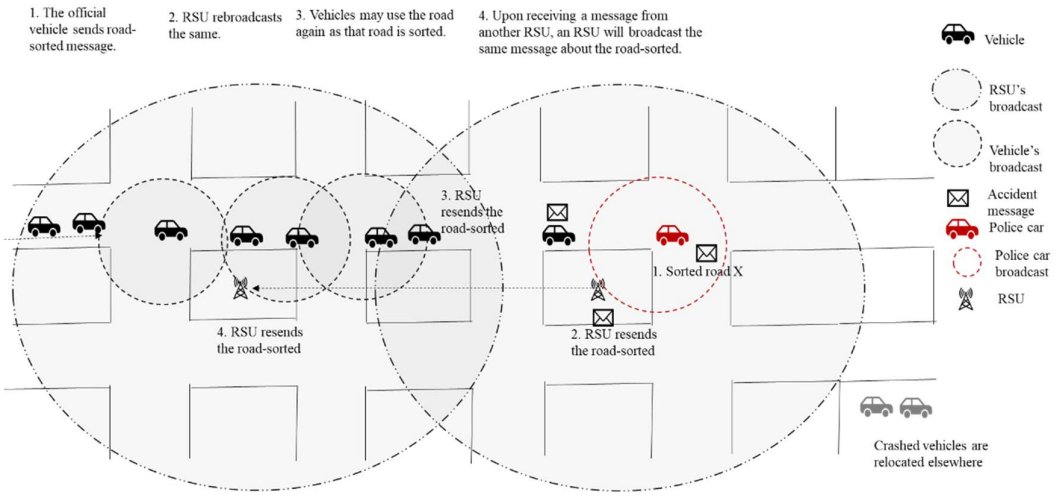


Fig. 4-9: Scenario of Sorting Out an Accident by a Police Car

An official vehicle can receive the accident message earlier than an RSU from a vehicle. In this case, it initiates the recovery process. An official vehicle that obtains the message first, retransmits it. If a nearby RSU receives it, it periodically broadcasts the "accident message on road X" within its transmission range so that other vehicles can avoid the road. Once resolved, the police car sends a "sorted-road" message which may reach an RSU through some intermediate vehicles. This RSU informs other nearby RSUs, and they also broadcast the traffic update periodically. This situation is depicted in Fig. 4-10, Fig. 4-11, and Fig. 4-12.

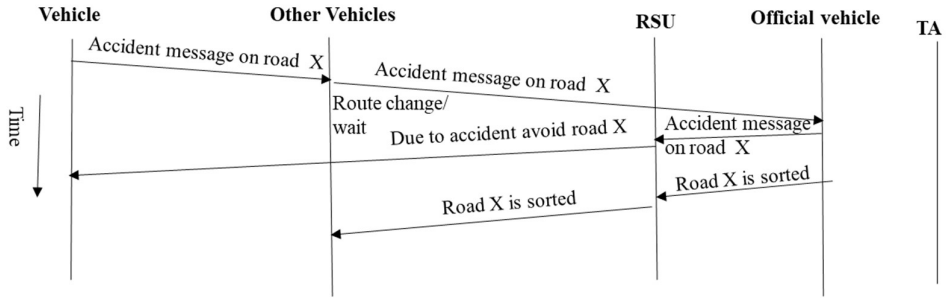


Fig. 4-10: Sequence Diagram of Dealing with an Accident Message by a Police Car

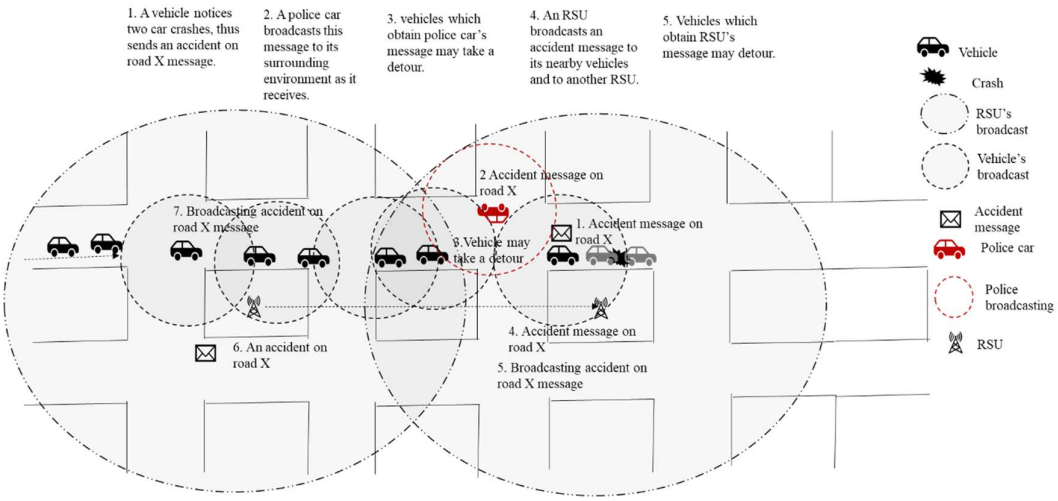


Fig. 4-11: Scenario of Accident Message Reception by a Police Car Before an RSU

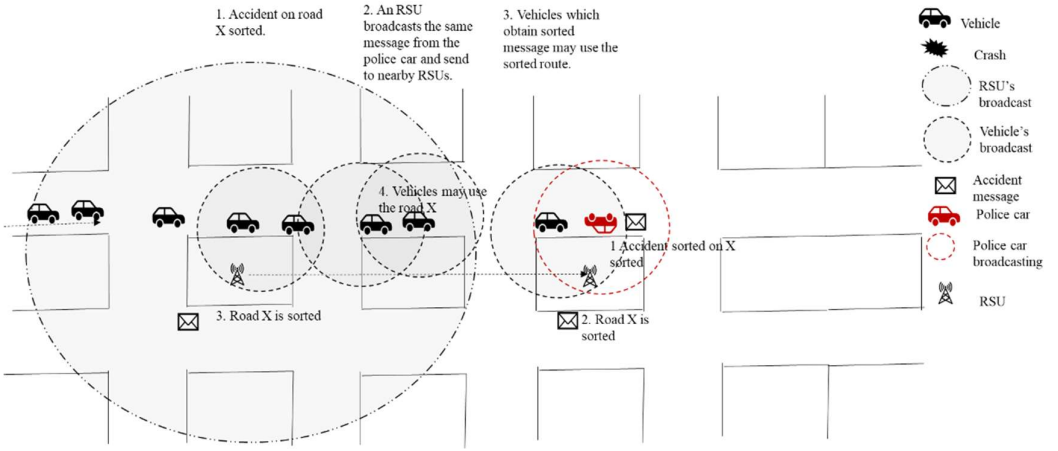


Fig. 4-12: Scenario of an Accident Event Sorted Out by a Police Car

4.3.2.1.5 Announcement of Congestion

A vehicle announces a "traffic jam on road X" message when its speed is less than 0.1m/s and is waiting more than 30 seconds on a road at one place and notices some vehicles are also queued before it. This message is relayed by intermediate vehicles and may reach an RSU. This RSU retransmits the "traffic jam on road X" message periodically towards normal vehicles within its transmission range until it receives a traffic clear message. The RSU also forwards the "traffic jam on road X" message to nearby RSUs to announce the same message in the VANET. In this way, more severe traffic jams can be avoided when an event is announced before it becomes severe. Vehicles that obtain this message may detour or wait in the queue. Later, when a vehicle observes the road is jam-free, it broadcasts a road clear (sorted-road) message. An RSU rebroadcasts the sorted-road message multiple times upon its arrival. The RSU that initially broadcasted the traffic jam message now announces a jam-free

message and sends it to nearby RSUs as they are also affected. Vehicles that receive the traffic updates may use the road "X" again. There is a different congestion message that is generated whenever the vehicle's speed is 1-13 m/s for more than 60 to 90 seconds. This message is handled in the same way as the traffic jam process. Fig. 4-13 and Fig. 4-14 show the message sequence diagram for traffic jam and congestion, respectively, and Fig. 4-15 and Fig. 4-16 show the corresponding scenario diagrams for broadcasting traffic jam / congestion messages and when the situation is resolved.

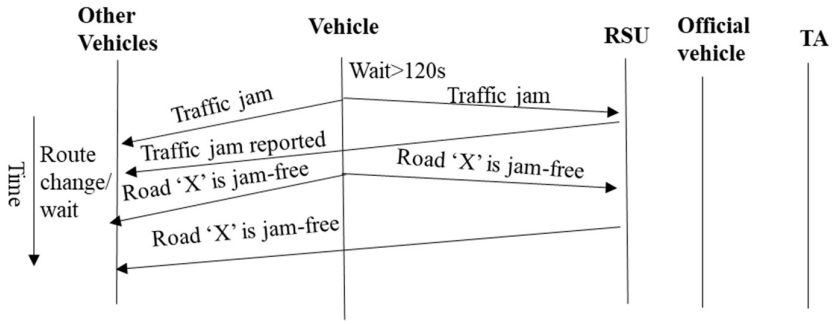


Fig. 4-13: Sequence Diagram of Reporting a Traffic Jam

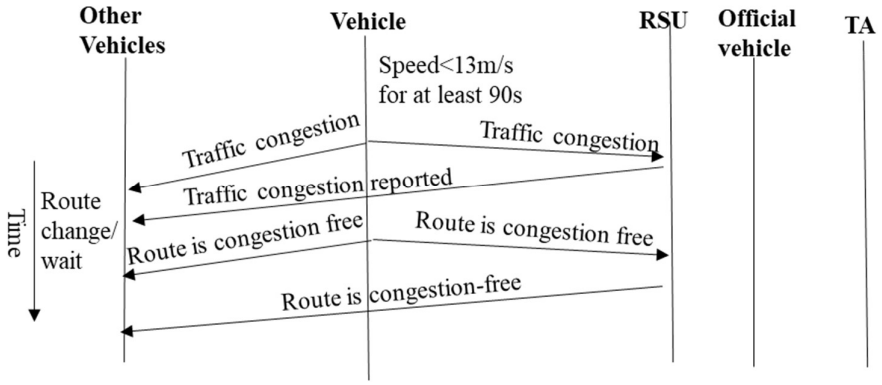


Fig. 4-14: Sequence Diagram of Reporting a Traffic Congestion

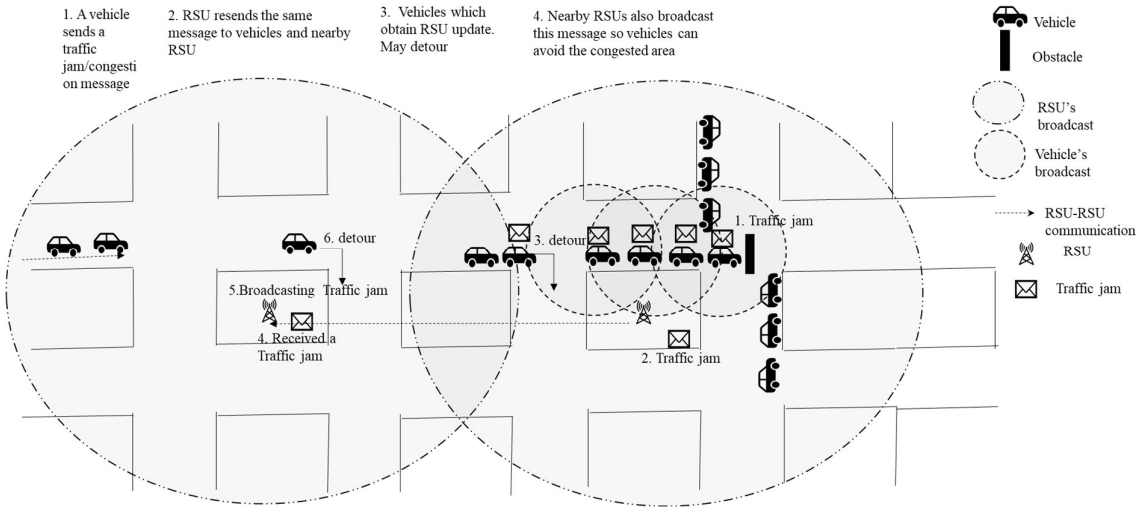


Fig. 4-15: Scenario of Reporting a Traffic Jam / Congestion

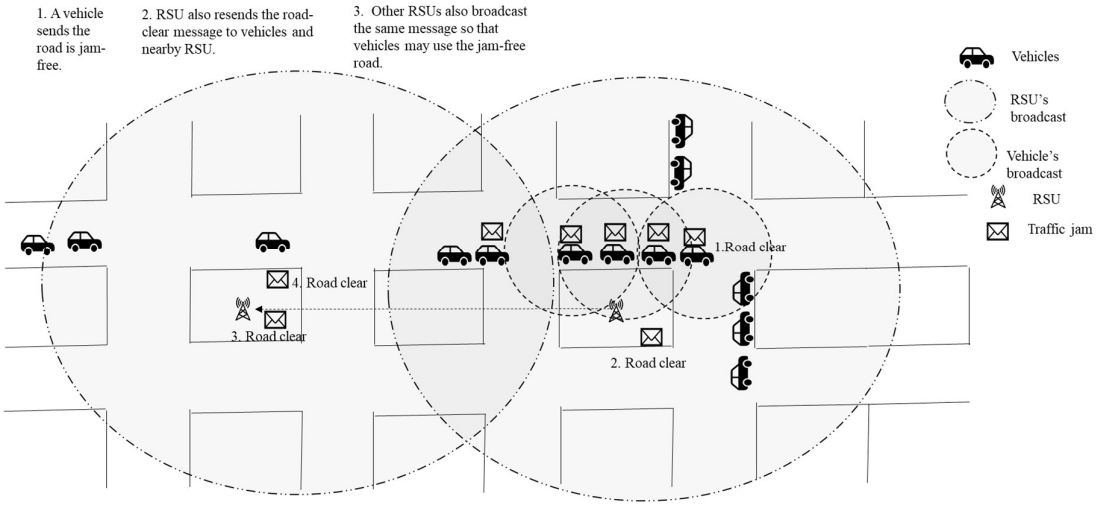


Fig. 4-16: Scenario of Broadcasting a Jam / Congestion Clear Message

4.3.2.1.6 Announcement of Obstacles and their Clearance

Suppose a vehicle notices an obstacle on road and then broadcasts a message saying, "an obstacle on road X". This message reaches one of the RSUs through intermediate vehicles. The RSU retransmits the same message in its transmission range. The RSU also informs this nearby RSUs. By this time, an official vehicle may receive the message and reply to the RSU with an "attending this issue" message. Other RSUs also broadcast the same traffic update within their transmission range. Vehicles that receive this traffic update can take a detour if they wish. Later, assume a police car visits the place where the problem is reported and clears the obstruction. After that, the police car broadcasts a message to the RSU by saying "no obstacles on road X". The RSU forwards it to nearby RSUs. Then the RSUs broadcast that the obstacle on road X is cleared now so that vehicles that are nearby can use that road

again. Fig. 4.17 shows a sequence diagram for this situation. Fig. 4-18 and Fig. 4-19 depict the corresponding scenario.

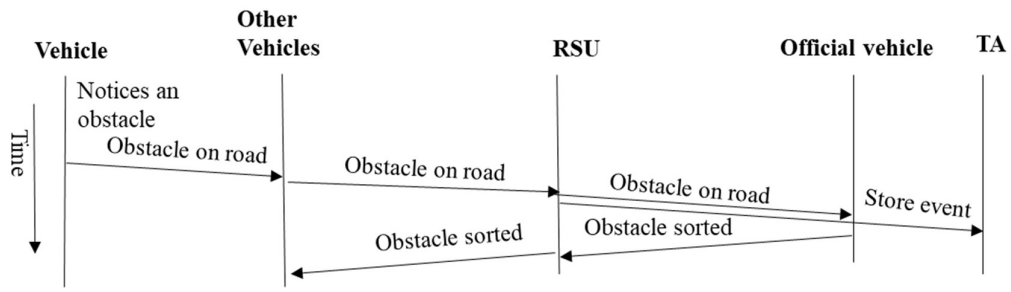


Fig. 4-17: Sequence Diagram of Reporting an Obstacle Message

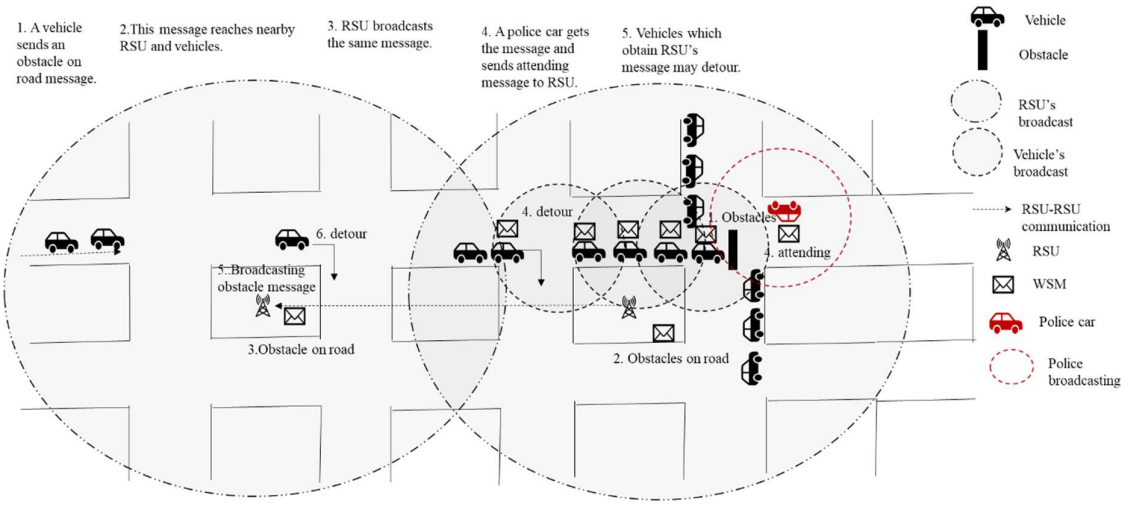


Fig. 4-18: Scenario of Announcing an Obstacle on Road Message

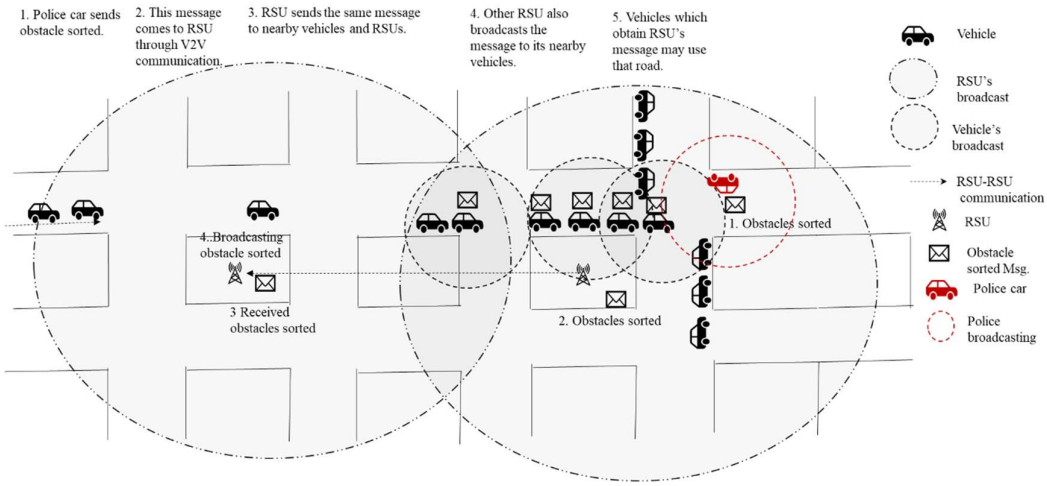


Fig. 4-19: Scenario of Broadcasting an Obstacle on Road Clear Message

4.3.2.1.7 Announcement of a Diversion

An official vehicle broadcasts a "diversion on road Y" message when the authority does not want vehicles to use a road. When an RSU receives this message, it informs nearby RSUs. RSUs periodically broadcast the same message obtained from the official vehicle to nearby vehicles within their transmission range. Vehicles that obtained this message may take a detour to reach their desired location hassle-free. Fig. 4-20 shows a sequence diagram for the diversion on road and Fig. 4-21 depicts the process of an announcement of diversion on road in the VANET.

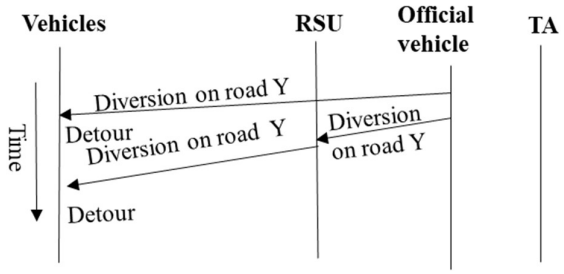


Fig. 4-20: Sequence Diagram of Broadcasting a Diversion Message on Road Y

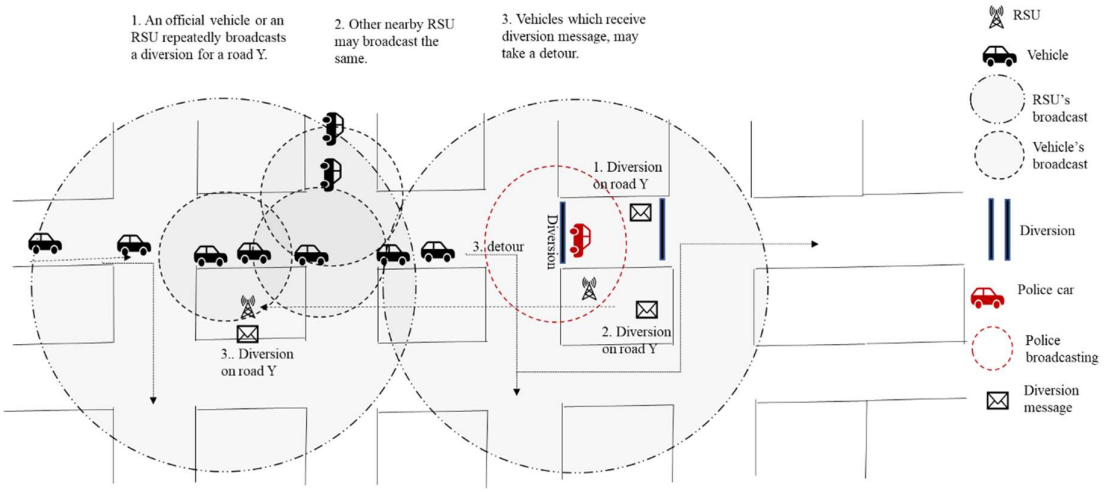


Fig. 4-21: Process of Broadcasting a Diversion Message on Road Y

4.3.2.1.8 Announcement of Stranded Vehicle

A vehicle that notices a stranded vehicle on road X broadcasts a message called "stranded vehicle on road X" to other vehicles and the RSU. The RSU forwards this message to a police car and other RSUs. A police car replies to this message upon reception and says that it is attending the event. Meanwhile, the RSU rebroadcasts the message. Later, the RSU receives a road-sorted message from the police car that earlier visited the place. Next, the RSU retransmits the road-sorted message within its transmission range and informs its nearby RSUs. Other neighbouring RSUs also retransmit the "sorted-road" message within their transmission range. Fig. 4-22 shows a sequence diagram of a stranded vehicle on a road message. Fig. 4-23 and Fig. 4-24 show the process of announcing a stranded vehicle on the road message in the VANET.

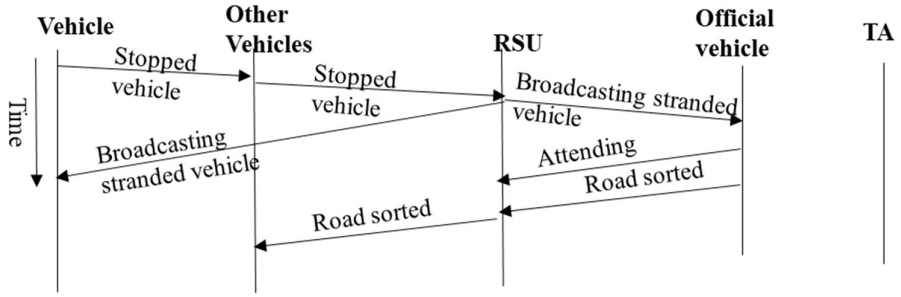


Fig. 4-22: Sequence Diagram of Broadcasting a Stranded Vehicle Message on a Road

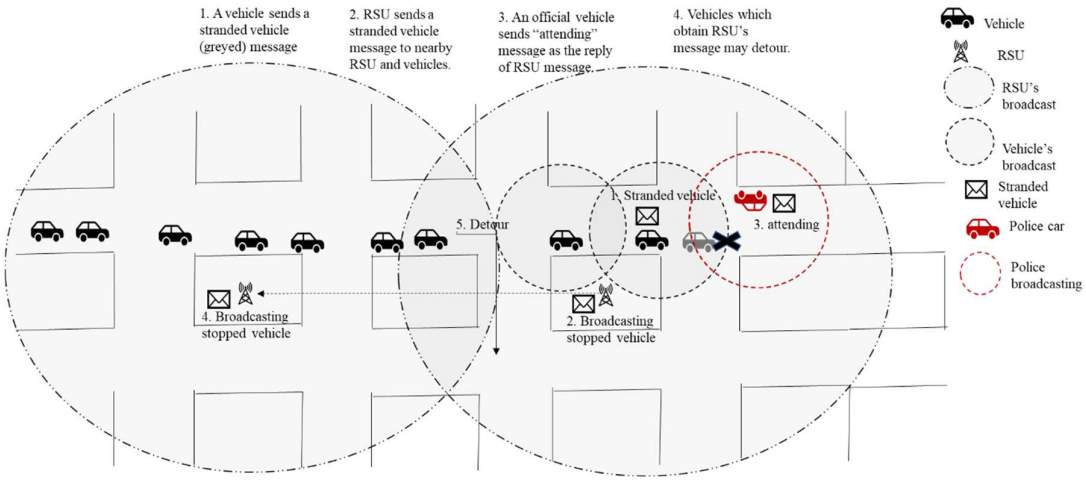


Fig. 4-23: Process of Broadcasting a Stranded Vehicle Message on Road X

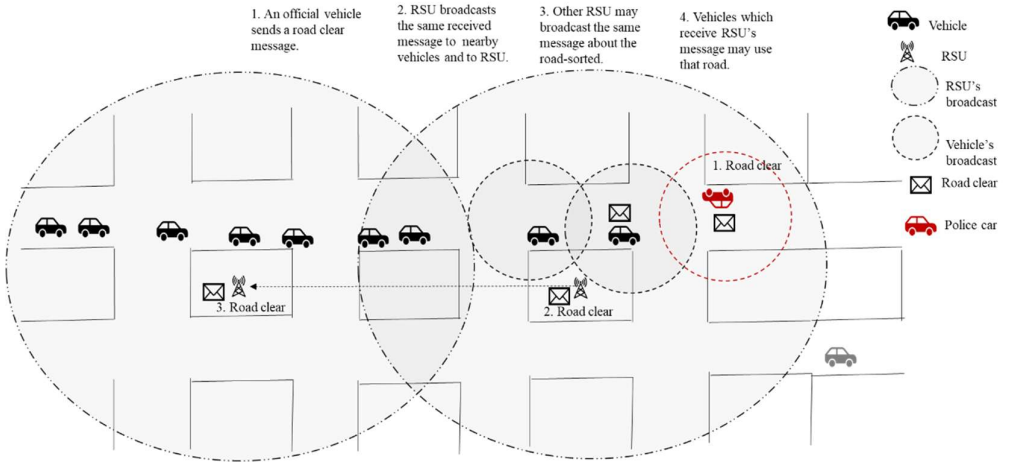


Fig. 4-24: Process of Broadcasting Stranded Vehicle Clear Message on Road X

4.3.2.1.9 Untrue Event Reporting

Suppose a vehicle observes an accident on road "X" and then broadcasts a message saying "accident on road X" to other vehicles within its transmission range. One follower vehicle detects that there is no accident at the claimed location and then broadcasts an untrue message about the accident on road X. The VANET has two conflicting messages. Thus, the RSU initiates a collaboration with trusted vehicles and official vehicles that are near the disputed event. Meanwhile, a police car nearby the reported accident sends there is no accident on road X. The RSU is now clear from the police car message that the sender sent an untrue attack. The sender who sends the accident message is punished by the RSU and the reporter of the untrue message earns an RSU reward. Fig. 4-25 shows a sequence diagram of untrue attack reporting and detection through collaboration and feedback from a police car. Fig. 4-26 depicts the process of untrue attack detection using collaboration from a police car.

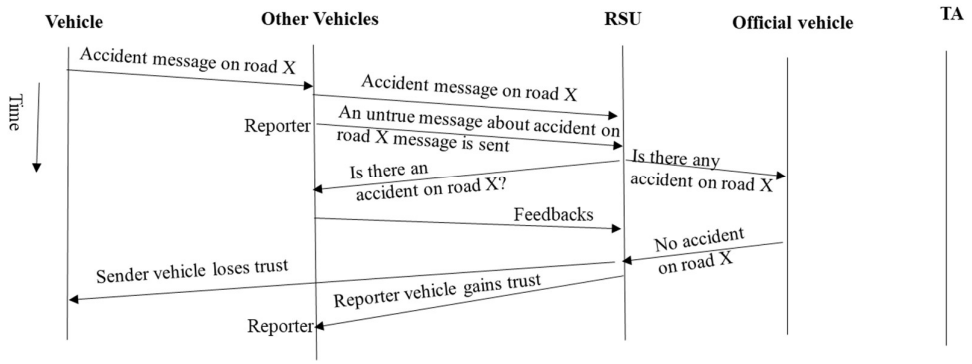


Fig. 4-25: Sequence Diagram of Detecting an Untrue Message and Issuing a Reward / Punishment

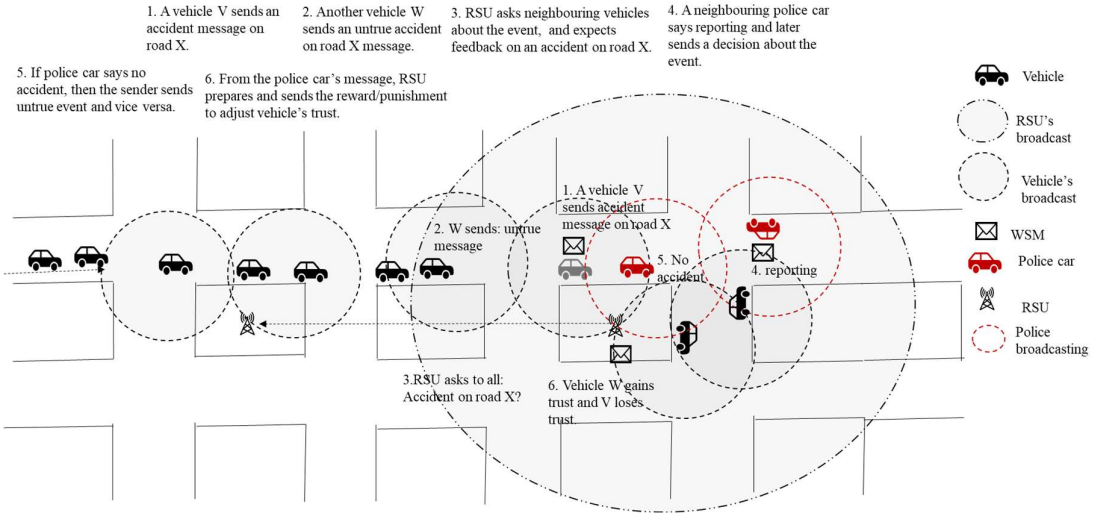


Fig. 4-26: Process of Untrue Message Detection and Punishing a Mischievous Vehicle

When a police car says that there is an accident on road X. Then the RSU finds the sender and sends a true accident message, and it gets an RSU reward. The reporter of the untrue attack receives an RSU punishment. This sequence diagram is depicted in Fig. 4-27. The process is depicted in Fig. 4-28.

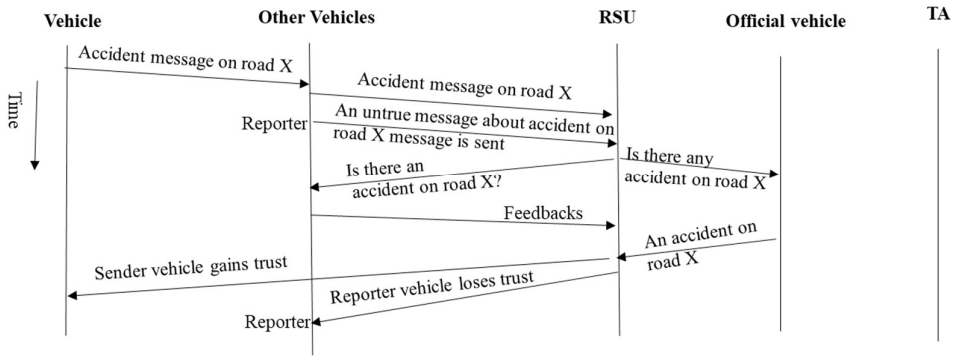


Fig. 4-27: Sequence Diagram of Detecting an Untrue Message with a Reply from a Police Car

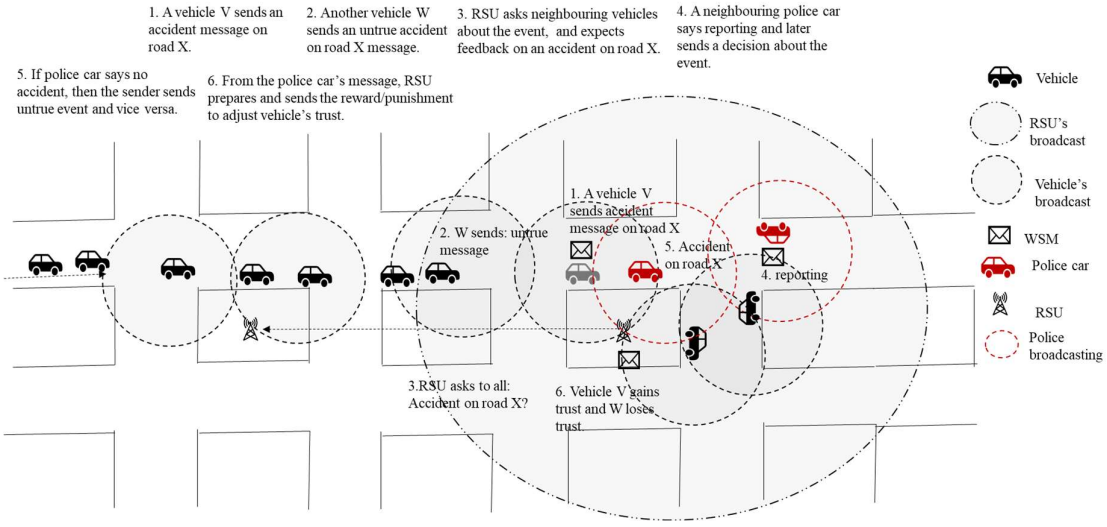


Fig. 4-28: Process of Detecting an Untrue Message with a Reply from a Police Car

If the collaboration involves both regular vehicles and official vehicles, then the RSU receives replies from official vehicles and regular vehicles. However, if a police car states it is far from the place where the accident took place, then the RSU chooses to wait for feedback from nearby regular vehicles until a timer expires. During this period, each vehicle may send its opinion relating to the event like "YES / NO and trust score". The RSU computes a sum of the product of opinions and trust to decide. A sequence diagram considering this scenario is shown in Fig. 4-29. The process of detecting an untrue attack with feedback from regular vehicles is illustrated in Fig. 4-30.

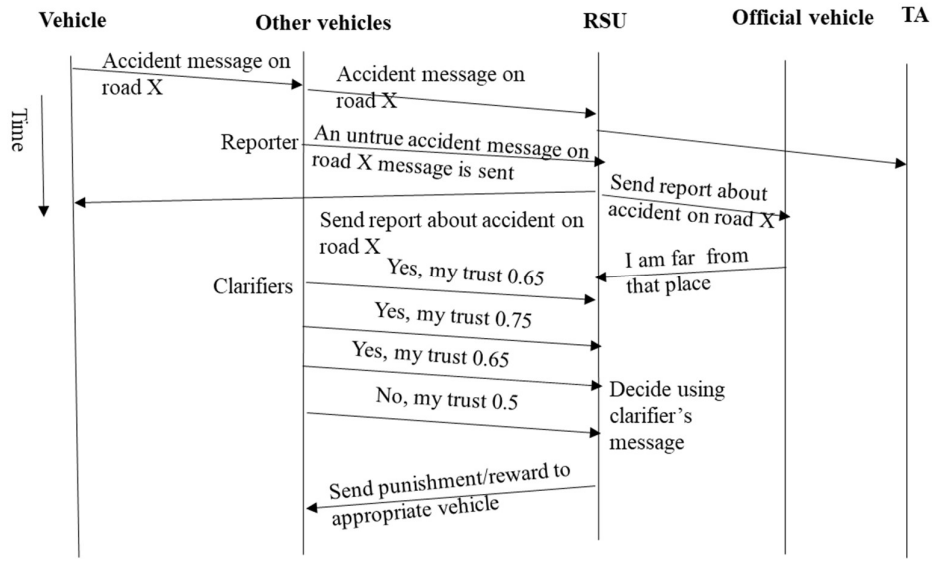


Fig. 4-29: A Sequence Diagram of Detecting an Untrue Attack from Feedback of Regular Vehicles

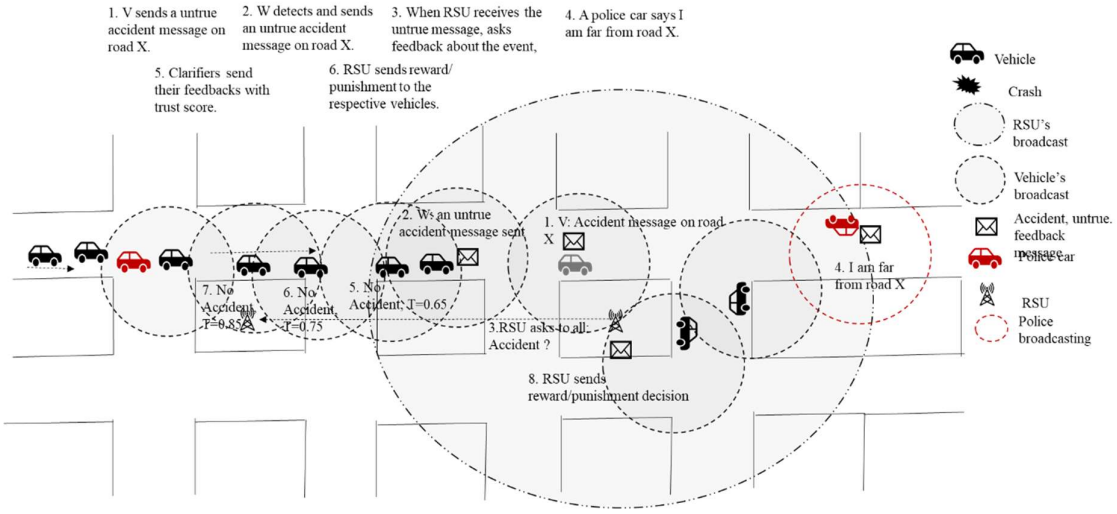


Fig. 4-30: Process of Detecting a Malicious Vehicle When it Launches an Untrue Attack

If whilst awaiting feedback from regular vehicles, it suddenly receives feedback from an official vehicle that the accident has occurred on road X, then the RSU bypasses the collaboration and resolves the dispute using only the feedback from the official vehicle. If the reporter lied, the RSU informs the reporter receives a punishment for false untrue attack reporting. Also, the sender receives an RSU reward. The sequence diagram for the scenario is depicted in Fig. 4-31. The process of dispute resolution is shown in Fig. 4-32.

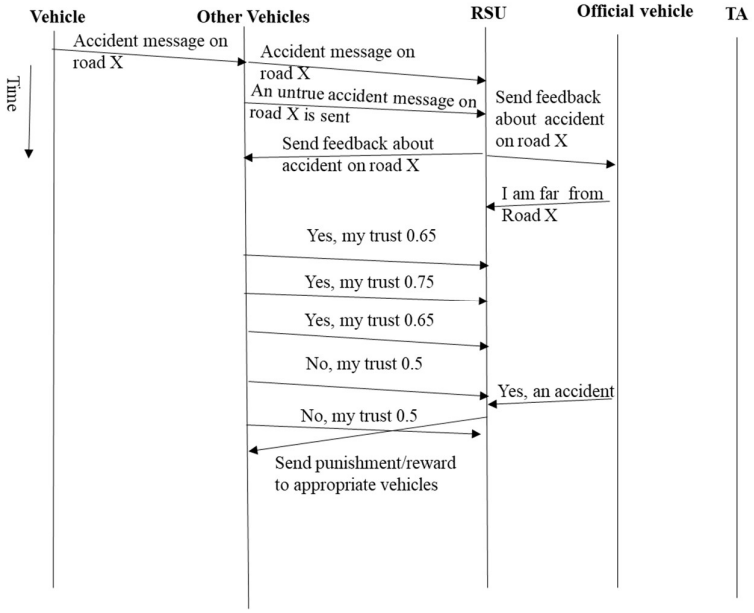


Fig. 4-31: A Sequence Diagram of Resolving Dispute from Feedback of Police and Trusted Vehicles

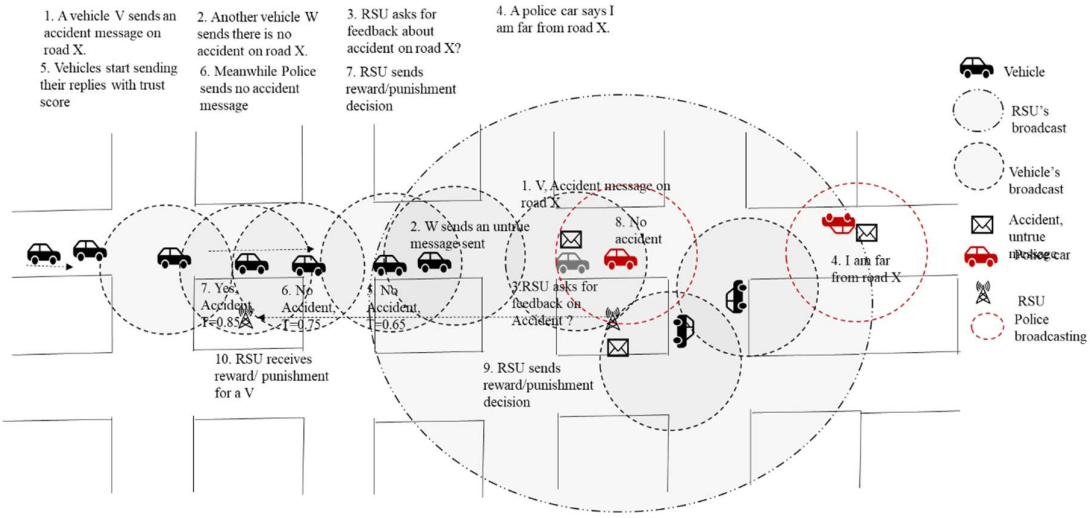


Fig. 4-32: Process of Dispute Resolution Using the Opinions of Police and Clarifiers

4.3.2.1.10 An Example Journey of a Vehicle Considering Different Events

Initially, vehicles and RSUs register themselves in the VANET using registration requests, and confirmation messages from the TA. They are also periodically broadcasting beacons which are not shown on the sequence chart in Fig. 4-33 as they are regular events. Suddenly a vehicle notices an accident on a road and announces this event in its vicinity which is relayed through intermediate vehicles and reaches an RSU and an official vehicle. After that, they work together to resolve the situation. After this, vehicles are running smoothly for some time. Later the vehicle finds itself in a

traffic jam that it announces. An RSU also receives the message from intermediate vehicles and retransmits the same to reduce the magnitude of the traffic jam. When the traffic jam ends, a vehicle announces “no traffic jam” that reaches the nearby RSU. This RSU broadcasts this message to vehicles that the road is free again. A vehicle then finds an obstacle on road, so it announces it, and an official vehicle comes to clear the road based on the announcement. When the road is clear a sorted-road message from the official vehicle informs an RSU that the affected road is resolved so the RSU retransmits the message multiple times. A vehicle then observes that the road condition is poor. So, it announces a poor road-condition message so that the following vehicles are aware of the situation. An RSU receiving this message warns vehicles to avoid possible adverse traffic events. An RSU can inform the TA to raise this matter with the RTA to repair the road. Similarly, a vehicle may locate a service via an RSU. After this, the green area shown on the sequence diagram signifies the VANET is running without any events. Finally, a vehicle is shown to experience a very low trust score. The TPD generates an access-blocking request to the RSU to send it to the TA. The TDP receives a confirmation of access-blocking from the TA to implement blocking of the concerned driver.

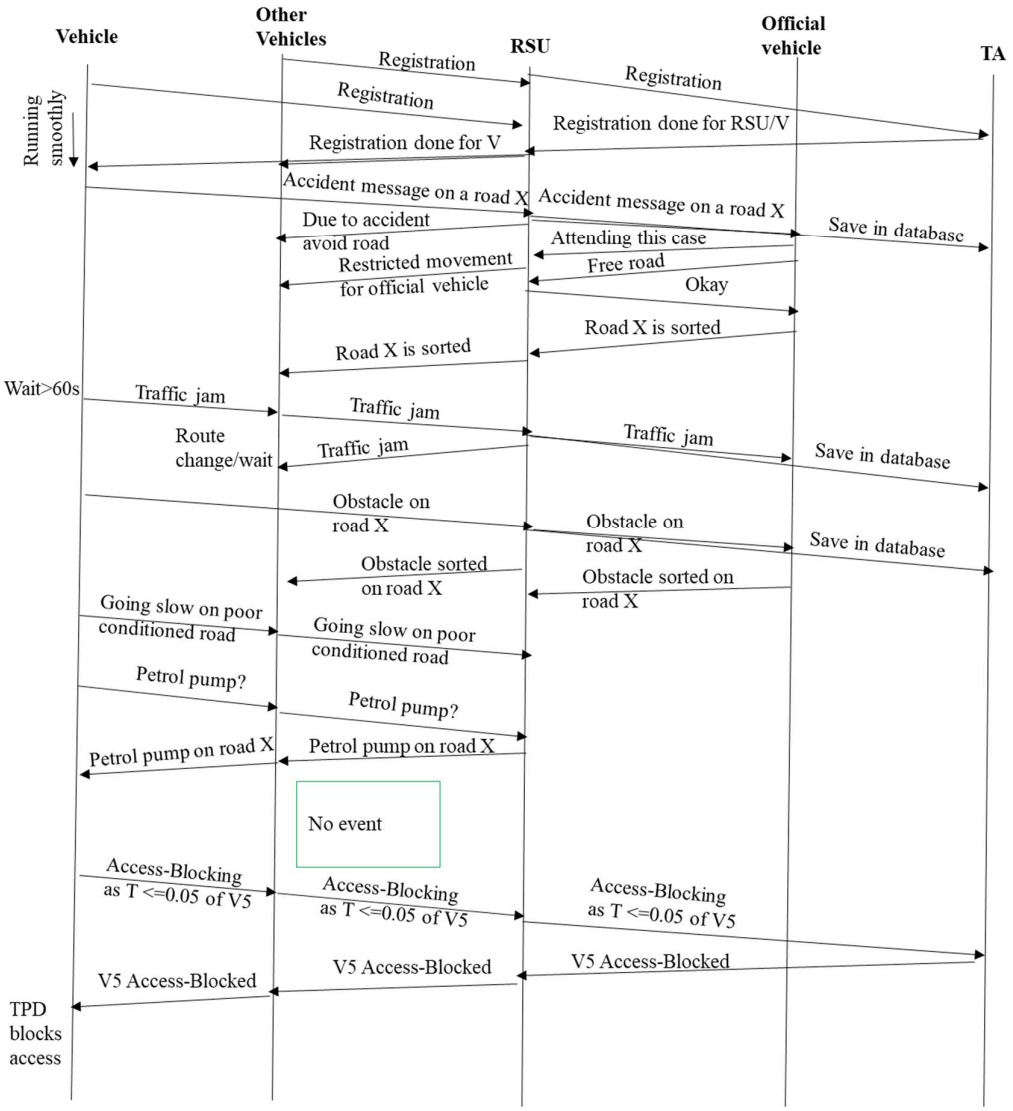


Fig. 4-33: A Series of Events Announcement from a Vehicle in a VANET

4.4 The Need for Trustworthy Message Announcements

4.4.1 Scenario 1 – Effect of Untrue and Trustworthy Message Announcements on Average Travel Time

Two experiments have been conducted using an alternate route scenario (Fig. 4-1b). One primary route is considered where the simulated vehicles run normally. Vehicles can also detour to a second route when they receive an announcement about an event on the primary route. In these experiments, only two RSUs and one TA are considered. The experiments are conducted in the presence of 10, 30, and 50 vehicles. First, the average travel time of all vehicles on the primary route is calculated during the normal running mode which is denoted by the blue-coloured line in Fig. 4-34. Then they are compared

with the average travel time of all vehicles when an untrue message is announced by vehicle V0 which is denoted by the orange-coloured line in Fig. 4-34. It is clear from Fig. 4-34 that the average travel time of all vehicles is increased by at least 20 seconds in all cases as vehicles are rerouted to the longer second route due to the untrue announcement. Due to an untrue announcement, some vehicles believed the message and take a detour. This does not happen always when the vehicle density is high, and they are inserted periodically. Some vehicles which are inserted later find the primary route free and use it. When the density is 30, most of the vehicles are near the junction, so they make the detour upon the message arrival. When there are 50 vehicles, some vehicles are at the junction, so they take the instant detour due to the untrue announcement. Later, other vehicles arrive at the junction, and they find the primary route free, so they do not detour to the alternate route. This is why the orange-coloured line shows a small decline in average travel time with 50 vehicles compared to the case when the experiment is conducted with 30 vehicles.

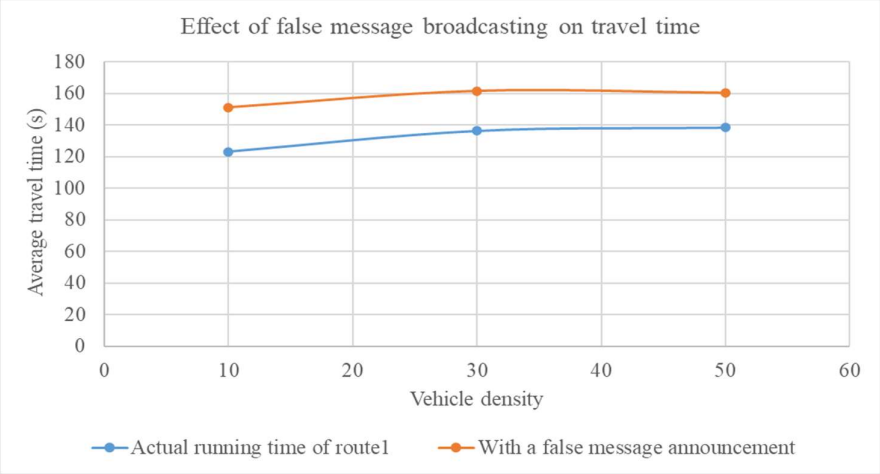


Fig. 4-34: Effect of Untrue Event Announcement on Travel Time

In the second experiment, an unannounced true event on the primary route causes the vehicles to be queued for 120 seconds. This is why, their average travel time is increased by at least 90s more than the normal case. This is depicted with the blue-coloured line in Fig. 4-35. However, these values are reduced by at least 36s using a trustworthy message announcement from V0 as shown by the orange-coloured line in Fig. 4-35. For vehicles that do not receive a timely traffic update, their travel time remains the same in both cases.

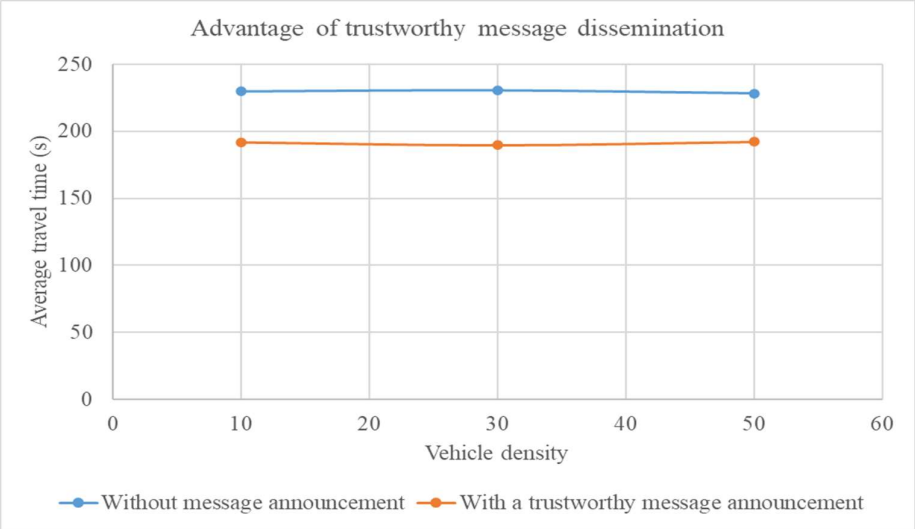


Fig. 4-35: Improved Travel Time with True Event Announcement

4.5 Verification

The framework is verified in the presence of malicious and benign behaviours of trusted vehicles. Before describing the details of the verification, first we discuss the verification of some of the existing trust models. The reputation model (Mühlbauer et al [14]) verifies its resilience against betrayal and inconsistent attacks. The reputation model assigns a reputation from 0 to 10 to vehicles and conducts a set of simulation which runs for 10000 simulation seconds. While thwarting betrayal attack, malicious nodes announce false events during a specific time interval only. Initially, they start their journey with a reputation of 5 and achieve reputation of 10. Then their reputation falls to 0 during this interval and then they build their reputation up once again because redemption is allowed even if a malicious vehicle reaches a reputation of 0. In contrast, in another experiment, they thwart inconsistent attacks. In this experiment, malicious vehicles initially earn a reputation of 9. Then they send both true and false messages alternatingly from a specific time to the end of the simulation. During this attack period, the reputation of malicious vehicles remains very low. In contrast, Zhou et al [116] verify direct and indirect trust evolution by simulating their trust model in MATLAB and they conduct two different experiments. In the first experiment, they only evaluate the direct trust from historical events. Eighty vehicles participate and five different security events are considered which have different weights. From this experiment, they plot the security degree in charts. One chart shows the security degree (trust) distribution of all nodes, and another chart illustrates the security degree of all nodes in a histogram. After calculating the security degree of all nodes, the minimum acceptable security degree is used to identify the trusted nodes from this experiment. In another experiment, they evaluate whether current nodes accept a new node or not. This is conducted by collecting recommendations from twenty vehicles and calculating their indirect trust. When evaluating a new node, all nodes send their recommendations and then the correlation coefficients of all recommendations are computed. It is found that the

correlation coefficients of only four vehicles have a lower value than 0.6 out of the twenty vehicles. Thus, there is a higher chance that these four vehicles will be malicious if the correlation coefficient is set at 0.6. They can reject new vehicles from joining the network deliberately.

To verify the proposed trust model, this experiment is conducted at least thirty times for 5000 simulation seconds with 10-100 vehicles. Two results from these experiments are depicted in Fig. 4-36 and Fig. 4-37 to illustrate the framework is successful in trust management in the presence of attacks. The horizontal axis represents the simulation time in seconds, and the vertical axis represents the trust score. All vehicles start with a trust score of 0.9. The reward is fixed for a single announcement or RSU interaction which is set to 0.08. and RSU punishments for three untrue announcements are set to 0.1, 0.3, and 0.5 (applying IPP) consecutively. Receivers report an event with a probability of 40% and the event supporting probability P from clarifiers is set to 20%. Vehicle V0 sends messages at 200s intervals starting from 100s.

4.5.1 Thwarting Consecutive Untrue Attacks and Access-Blocking

As the simulation begins, V0 sends an untrue message every 200s starting from the 100s until it is access blocked. This test demonstrates the effect of announcing three consecutive untrue messages within the proposed framework. The framework implements the access-blocking of V0 when it announces three consecutive untrue messages. RSUs find that V0 has sent untrue messages at 100s, 300s, and 500s consecutively and hence V0 receives punishments by 0.1, 0.3, and 0.5 consecutively at 220s, 620s, and 820s which are marked by the steep fall in the chart. The punishment amount is selected to block access to a highly trusted vehicle with a trust score of 0.9 after performing three untrue attacks (for example, if the three punishments are 0.1, 0.3, and 0.5, then the total punishment will be 0.1+0.3+0.5=0.9). Therefore, the vehicle would be left with a zero trust score; however, the framework limits the lowest trust score to 0.05. After the third untrue message from V0, the RSU immediately sends an access-blocking request message to the TA to request the access-blocking of V0 in the network. Then the TA sends an access-blocking confirmation to the neighbour RSUs of RSU from which it receives the last punishment. The TPD implements access-blocking immediately. The other vehicle V3 is consistently benevolent throughout the simulation period and hence its trust score remains constant. In this experiment, other vehicles participated but are omitted from the chart for clarity.

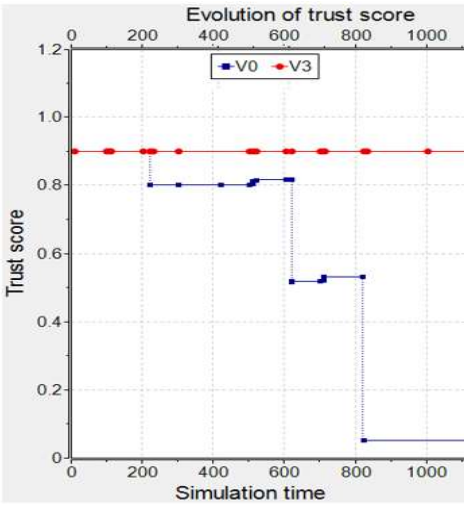


Fig. 4-36: Thwarting Repeated Untrue Attacks and Access-Blocking of V0 at T=0.05

4.5.2 Thwarting Inconsistent Attacks and Access-Blocking

The framework is verified in the presence of malicious and benign behaviours of trusted vehicles. To this end, this experiment is conducted thirty times for 5000 simulation seconds with 10-100 vehicles. One result from these experiments is depicted in Fig. 4-37 to illustrate trust management in the presence of attacks. The horizontal axis represents the simulation time in seconds, and the vertical axis represents the trust score. All vehicles start with a trust score of 0.9. The reward is fixed for a single announcement or RSU interaction which is set to 0.08. and RSU punishments for three untrue announcements are set to 0.1, 0.3, and 0.5 (applying IPP) consecutively. Receivers report an event with a probability of 40% and the event supporting probability P from clarifiers is set to 20%. Vehicle V0 sends messages at 200s intervals (simulation seconds) starting from 100s. Fig. 4-37 records the inconsistent behaviour of V0 with the consistent behaviours of V1 and V2.

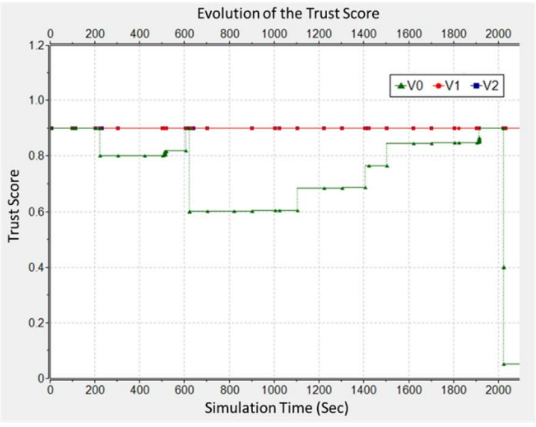


Fig. 4-37: Thwarting Inconsistent Attacks and Access-Blocking of V0 at T=0.4

When V0 sends untrue messages, the RSU punishes it by 0.1 and 0.3 consecutively at 220s and 640s. Conversely, when V0 announces trustworthy messages consecutively at 500s, 1100s 1300s, 1500s, 1900s, the TPD adds a reserve reward at 620s, 1220s, 1420s, 1620s, and 2020s; these rewards are withheld for 120s. V0 receives complaints for message announcements at the 700s, 900s, and 1700s. Thus, the TPD does not add any reward for these announcements. After this, the RSU punishes V0 by 0.5 which is shown by a large reduction in the trust score at 2020s followed by an access-blocking message which sets its current trust score to 0.05 irrespective of whatever (T=0.4) it previously had. In this way, V0 is blocked from network access by the framework. From this simulation result, it can be concluded that the Incremental Punishment Policy (IPP) will demotivate vehicles from attacking repeatedly. The IPP provides a flexible means of punishing and blocking vehicles for their inconsistent behaviours although they may sometimes announce trustworthy messages in between their malicious activities. As vehicles receive higher punishment in each subsequent untrue announcement, vehicles with inconsistent behaviour will be isolated as well. Additionally, we allow only three malicious actions within the simulation timeframe (for example 5000s) from a trusted driver to become blocked to limit further harmful actions. So, in summary, this trace represents an example confirming the system can successfully detect inconsistent behaviour of a malicious vehicle and punish it accordingly.

4.6 Performance Evaluation

In this section, a performance evaluation of the proposed approach is performed. These experiments include but are not limited to accuracy measuring in the presence of both true and untrue event announcements and a comparison of the response time and the communication overhead between this framework and a baseline approach given in Mühlbauer et al [14]. The baseline approach is a receiver-side reputation approach which is compared with the proposed trust model in Table 4-2.

Table 4-2: Comparison between the proposed trust model and baseline approach (Mühlbauer et al [14])

Feature	Proposed trust model	Baseline reputation model
Evaluation point	Sender side	Receiver side
Trust update point	At the TPD of vehicle	At RSU
Timer needed for verification an event?	No, receiver believes instantly unless one of the receivers visit and found the event is not in place, then issues an untrue attack in the network.	Yes, receiver waits and collects many broadcasts from many vehicles, and then decide using majority voting, weighted voting, maximum reputation, always positive and always negative.
Announcement consideration	One sender originates, others relay the message.	Many independent senders originate new event announcement.
Trust update mechanism	Reward, punishment	Positive or negative feedback counter to manipulate reputation.
Fuzzy logic-based reward or punishment scheme	Yes	No
Use of infrastructure	Yes	Yes
Multiple driver trust management	Yes	No

Driver behaviour modelling and analysis	Markov chain-based state diagram	No
Resilience to attacks	Untrue, Inconsistent, Cooperation attacks	Newcomer, Betrayal, and Inconsistent attacks
Dispute resolution	Upon an untrue attack reporting, RSU rules the validity of dispute by collecting responses from trusted vehicles and then takes a threshold-based decision using sum of product of trust and response.	No

The simulation model described in Section 4.3 is used in the presence of varying traffic densities. Clarifiers generate varying percentages of malicious and benevolent feedback to classify events as true negative, true positive, false positive, and false negative observational data. Analysis shows the proposed framework can classify events as expected. This means that when there is more benevolent feedback the RSU can classify an event correctly and vice versa.

This set of experiments considers the generation of varying ratios of positive and negative feedback when classifying a disputed event. Moreover, we confirm the minor impact of vehicle density on the results as RSUs ignore repeated complaints regarding the same event. The RSU forwards the first complaint to nearby RSUs which avoids invoking costly concurrent collaboration procedures at other RSUs. Furthermore, the proposed framework is compared against a reputation approach (Mühlbauer et al [14]) in terms of response time and communication overhead. The response time is the decision time of receiver vehicles when they receive an event message in the network. A trust management model can be considered efficient when receivers can decide about an event in the fastest possible time relative to trust systems where additional computation and communication are required after the arrival of messages. Hence, response time is an important indicator of performance. Another useful metric is the communication overhead since a trust model with lower communication overhead reduces the burden of message transmission and processing. For this reason, a trust model with a lower response time and communication overhead can be regarded as superior to one where these values are higher. Furthermore, these two metrics affect the performance of the network communication, such as channel availability, hence the proposed approach is compared with a reputation approach that suffers from these two factors. Results from the analysis show the proposed framework outperforms the existing one as a receiver vehicle in the proposed framework can decide on the appropriate action without further communication within the VANET. However, the proposed framework requires broadcasting feedback additional to the traffic event if a reporter invokes an untrue attack event. Within the timeframe of this research, we have produced a framework that is able to thwart untrue and inconsistent attacks. In addition, one form of cooperation attack is theoretically analysed with the trust framework. However, in future, the framework can be extended to thwart additional security threats.

4.6.1 Measured Accuracy of the Proposed Framework

In this set of experiments, vehicles announce both true and untrue messages. Some vehicles also report these announcements. Thus, some disputes arise which need to be addressed to determine the validity of events. Because for an announcement, both "an event" and "no event" coexist. An RSU is required to resolve these disputes as they are close to the event and authoritative. RSUs collect feedback and classify events. Messages are classified according to their validity as follows:

- A true negative (TN) state arises when the sender sends a true event and the RSU correctly confirms the message is not an attack. This is also the case when an RSU has received an untrue attack message from a source and then discovers (through further investigation) that the original sender of this message sent valid information. In this instance, we say the sender "wins the dispute".
- A false negative (FN) state is when an RSU considers a message to be true when the original message is actually reporting untrue / false information (i.e., an attack). That means, the reporter that created the untrue message loses the dispute and is penalised.
- A false positive (FP) state is when an RSU identifies a sender message as an attack (i.e., the sender of the original message is deemed to have initiated a message of a false event) and the reporter, lies and sends an untrue message which is incorrectly considered to be true. However, the original sender's message is actually true.
- A true positive (TP) state is when an RSU identifies a sender message as false, and this is in fact the case. This situation would arise if the sender lied and sent incorrect information. One or more reporters then send conflicting information, and the RSU correctly resolves that the sender has lied, losing the dispute, and is penalised.

The following matrix is also used to classify cases throughout the simulations. Based on the RSU decision, the RSU judgements are one of the following from Table 4-3 as: TN, FP, TP, or FN.

Table 4-3: Results Classification Matrix

	Predicted true	Predicted false
A true event	True negative	False positive
A false event	False negative	True positive

4.6.1.1 Simulation Setup

These series of simulations have been conducted using the Erlangen city map from Veins. In this map, one predefined route is created which is used by both regular and official vehicles. Events are generated periodically at predefined times from some selected sender vehicles. The system starts with no vehicles in the terrain model and once a vehicle is added, it remains in the system until the simulation terminates. Once a predetermined number of vehicles enter the system, no more are permitted. The set of parameters for conducting the accuracy measure during the set of simulations is listed in Table 4-4. These parameters are selected in the following way. Some of the parameters are already defined in

Veins, for example, the Erlangen city map is 2.5km × 2.5km, and the centre frequency is 5.89 GHz. Also, some parameters are defined in the trust model. We need to evaluate the trust model in the presence of different traffic densities, RSUs, TA, multiple lanes, and with variations in speed. We have conducted 5 different trials for each vehicle density and event supporting probability because, during the simulation, we found the number of reported cases vary substantially so that we reduce this bias by averaging data from all trials. We have also used a warm-up period to delay starting periodic event announcements until all vehicles join the simulation. The transmission range of vehicles is 300m as we found less collusion occurs with this setting. The collaboration timer is 120s as we found with provided a sufficient number of feedback messages within this duration while running different scenarios before the data collection process starts. We have allowed different events to have different ratios of malicious feedback so that the accuracy of the framework in different conditions can be presented.

Table 4-4: Parameters for Simulation Setup

Selected parameters		Value
Details about simulation	Simulation area	2.5km X 2.5Km
	Simulation Scenario	Erlangen city map, Manhattan grid, and alternate route scenario.
	Number of vehicles	[10, 30, 50, 70, 90, 100]
	Number of police vehicle	1 to 10
	Number of ambulances	1 to 5
	Number of fire service vehicle	1 to 5
	Number of RSUs	12
	Number of TA	1
	Number of routes	1
	Number of flows	1
	Speed of vehicles	Max 80 m/s
	Simulation time	4000s
	Transmission range	300m
	Centre frequency	5.89 GHz
	Data size	1024 bits
	Header length	80 bits
	Number of trials	5 for each case
	How is each sample collected?	After averaging all trials
	Warm-up period	700s
	Number of sources	3
	Periodic announcement	At 100s, this is configurable.
	Event supporting probability	[0, 0.2, 0.4, 0.6, 0.8, 1]
	TPD module	Inside every regular vehicle
	Initial trust	0.8
Trust management framework details	Maximum trust	0.9
	Minimum trust	0.06
	Trust score to access-blocking	0.05
	Reward withhold timer	120s
	Adaptive reward	Based on the promptness of the driver and honest behaviour
	Three malicious events (3ME) count for the driver	Initiate access-blocking procedure.
	Collaboration timer	120s
	Untrue attack	From regular vehicles
	Inconsistent attack	From regular vehicles

Attacker Model	Disputes decider	RSU

We repeat each experiment five times to collect trial data for each vehicle density and for every probability of supporting an event. This sample data is then averaged for analysis. We use the probability P to control the support or denial of events from clarifiers through YES / NO responses. For example, with a probability of P=0, clarifiers always send NO. For P=1, clarifiers always send YES. For probabilities of P=0.2…0.8, clarifiers send YES / NO responses accordingly. In this way, the analysis considers varying ratios of benevolent and malicious feedback. We arrange for senders to always announce true events in one set of experiments, whereas, in another set, they always announce untrue events. One or more reporter vehicles may disseminate an attack in the VANET where the attack dissemination is randomized with a probability of 0.4 though this can be configured to any value from 0 to 1. If this value is set to a lower one, vehicles do not complain about the original announcement most of the time. As the complaints are based on randomness, so the number of reported events varies in each trial. Receiver vehicles forward these events to RSUs. The RSU then initiates the "collaboration" process to determine what has actually happened in regard to a particular event. Based on the collected data, a disputed event will be classified as either TN or FP or TP or FN. If an RSU receives feedback from an official vehicle, the RSU uses it directly to decide on the event and the RSU collaborative process is bypassed. Vehicles are selected from the beginning, middle, and end of the flow for message announcements. For example, when the number of vehicles is 50, then VO, V24, and V49 are preselected senders of events. The following explanation considers P, the probability of feedback generation, and D, the vehicle density.

4.6.1.2 Analysis of Results

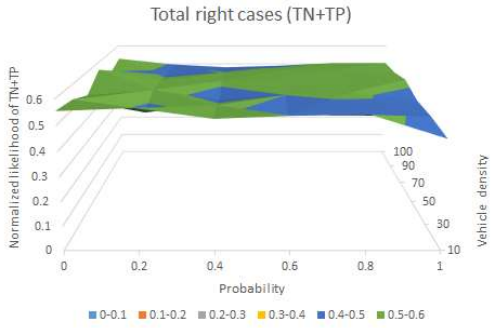
In Fig. 4-38, the x-axis represents vehicle density, the y-axis represents the probability of being truthful or not, and the z-axis represents the normalized likelihood of classified cases. We define the normalized likelihood of TN / FP classified cases as the ratio of the average number of classified TN / FP cases to the average number of reported events that RSUs classify as TN / FP using the dispute resolution mechanism. When the vehicle density increases, the number of reporters also increases. However, throughout the simulations, increasing vehicle density is shown to have only a marginal impact on the results as the RSUs ignore repeated reports/complaints concerning the same event from multiple reporters. This is possible as the RSU which receives the first complaint concerning an event forwards a notification of it to other RSUs in the vicinity to prevent invoking further costly and redundant collaboration procedures.

Fig. 4-38a shows the true negative results for a series of simulations where sender vehicles only announce true events. Overall, as P increases, the possibility of classifying TN cases also increases. This means, the framework correctly classifies disputed events if most clarifiers send trustworthy

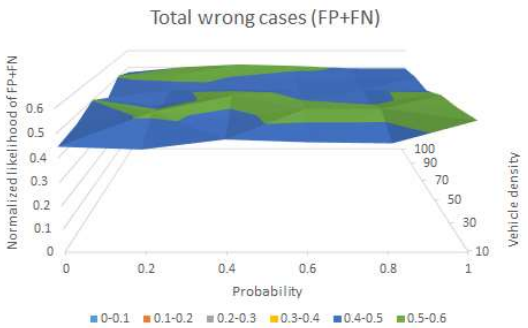
feedback. As expected, at P=0, there are no TN cases because all clarifiers have denied the original events (we can describe this situation as all clarifiers sending malicious feedback). Alternatively, at P=1, all reported events are detected as TN since all clarifiers only send YES (we can describe this situation as all clarifiers only sending trustworthy feedback). The TN cases increase rapidly from P=0.4 to P=0.6 as the proportion of received YES responses is sufficient to support the sender announcement at RSUs. Also, the number of TN cases increases with the rise of vehicle density as the increased traffic supports the sender's announcement. When P changes from 0.2 to 0.4 or 0.6 to 0.8, the TN cases increase significantly as truthful clarifiers have also increased in both situations. However, the graph does not show any major impact of increasing vehicle density as RSUs ignore additional complaints relating to the same event.

Fig. 4-38.b shows the false positive chart for a series of simulations when the sender vehicle announces only trustworthy messages. Overall, when P increases, the possibility of detecting FP cases decreases so that the graph shows a decreasing trend. This means the proposed framework incorrectly classifies the disputed events as most clarifiers deny the originated events. At P=0, at every density, all cases are FPs as all clarifiers send only NO (we can consider this situation as when all the vehicles send malicious feedback simultaneously). Alternatively, there are no FP cases at P=1 as all clarifiers send only YES. This means all classified cases are TN. The rapid fall in FP cases has been noticed from P=0.4 to 0.6 as the feedback pattern has flipped with more YES responses. This means the proportion of received YES responses at RSUs with P=0.6 is sufficient to support the sender announcement more than at P=0.4; thus, they are treated as TN cases. The FP cases decline significantly from P=0.2 to 0.4 and from P=0.6 to P=0.8 as RSUs receive additional YES responses from clarifiers with increases in P value (which supports the sender announcement more so). However, the graph does not show any major impact of increasing vehicle density as RSUs ignore additional complaints relating to the same event.

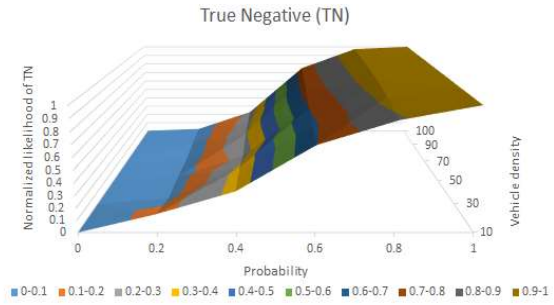
Fig. 4-38.c shows the TP chart for a series of simulations when the sender vehicle announces only untrue messages. Overall, when P increases, the possibility of detecting an untrue event as an untrue event decreases so that the graph shows a decreasing trend. That means the proposed approach can correctly classify the untrue events if most clarifiers deny the originated events with NO feedback. Hence, at P=0 at every density, all cases are TP as all clarifiers send only NO. We can consider this situation as all clarifiers only sending trustworthy feedback to the RSU. Alternatively, TP cases are lowest at P=1 for every density as all clarifiers reply YES. This can be described as the situation when all clarifiers send malicious feedback. Very few cases are detected as TP with P=0.8. The TP cases decline significantly from P=0.2 to 0.4 and P=0.6 to P=0.8 as the RSU receives fewer NO responses (i.e. fewer vehicles deny the senders announcements) from clarifiers with increasing P value. A rapid fall in TP cases is noticed from P=0.4 to 0.6 as the feedback pattern has changed with more YES responses. This means that these additional YES responses from clarifiers support the sender's announcements with increases in P value; this reduction is seen at P=0.6 from P=0.4.



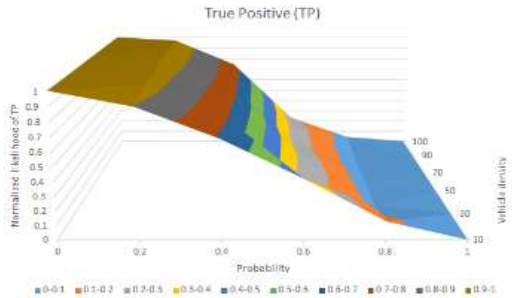
a) Total Right Cases



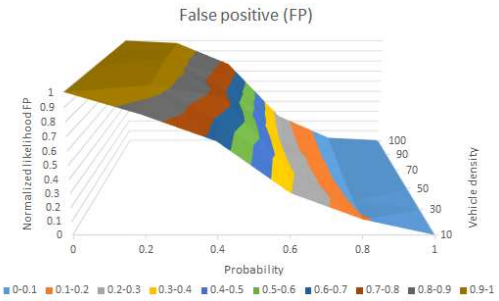
b) Total Wrong Cases



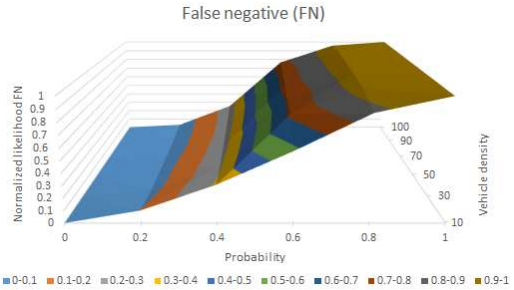
c) True Negative Cases



d) True Positive Cases



e) *False Positive Cases*



f) False Negative Cases

Fig. 4-38: Normalized Likelihood Classified Cases

Fig. 4-38.d shows the false negative chart for a series of simulations when the sender vehicle announces only untrue events. Overall, when P increases, the possibility of detecting an untrue event as a true event to the RSU is also increased so that the graph shows an increasing trend. That means the proposed framework incorrectly classifies untrue events as true events as FN when most clarifiers support the originated untrue announcements with malicious YES. Hence, at P=0, at every density, FN cases are the lowest as all clarifiers send only NO feedback. We can describe this situation as all clarifiers being honest when sending their feedback. Alternatively, at P=1, at all densities, all cases are

classified as FN as all clarifiers send YES. We can describe this situation as all clarifiers sending malicious feedback.

Overall, Fig. 4-38, shows some nonlinear increments/decrements when the P value changes. The TN / FN characteristic does not change linearly with the change in P values as the reception of YES feedback at various traffic densities is not always sufficient to classify all events as TN / FN. Thus, the normalized likelihood of classified TN / FN cases is lower at lower P values and is greater at higher P values than the expected trend. To classify an event as TN / FN, an RSU needs more YES / NO feedback than NO / YES, respectively. The proportion of YES and NO feedback received at an RSU is reflected in the decision and hence causes the curve to vary non-linearly with P. Similarly, the TP / FP curve does not show a linear relationship with increasing P values since the reception of YES feedback at various traffic densities is not always adequate to classify all events as TP / FP. Thus, the normalized likelihood of classified TP / FP cases is higher at lower P values and is lower at higher P values than the expected trend.

4.6.2 Performance Comparison of the Proposed Framework and the Baseline[14]

A baseline approach (Mühlbauer et al [14]) is implemented alongside the proposed framework to compare message flows. This evaluation shows that the receiver-side trust evaluation approach suffers badly from communication overhead due to trust metric dissemination as receivers are busy with trust verification after the arrival of messages. In the baseline scheme, the trustworthiness of a sender is decided using one of the following schemes: majority voting, weighted voting by reputation, and highest reputation level. The feedback is collated at the RSU, and the trust score is subsequently interrogated. For each event, feedback is collected and RSU sends updated trust to the vehicles. Alternatively, when an untrue attack is reported, then the RSU is required to run a timer and collect feedback. After that, the RSU sends the reward/punishment to the respective vehicles.

4.6.2.1 Simulation Setup

This set of experiments runs for 800 simulation seconds and is repeated 10 times to obtain the average number of messages exchanged in the presence of 10 to 70 vehicles. An event is introduced deliberately at 400 seconds in both approaches. In the baseline, all vehicles upon observing the event, announce it. Conversely, in the proposed framework the announcement of an event from one vehicle is adequate with receivers relaying it up to 4 hops.

4.6.2.2 Analysis of Results

In Fig. 4-39, the x-axis represents the number of vehicles present in the simulation, and the y-axis represents the communication overhead for a single event. This framework is compared against the approach in Mühlbauer et al [14] with 30 and 45-second interval timers.

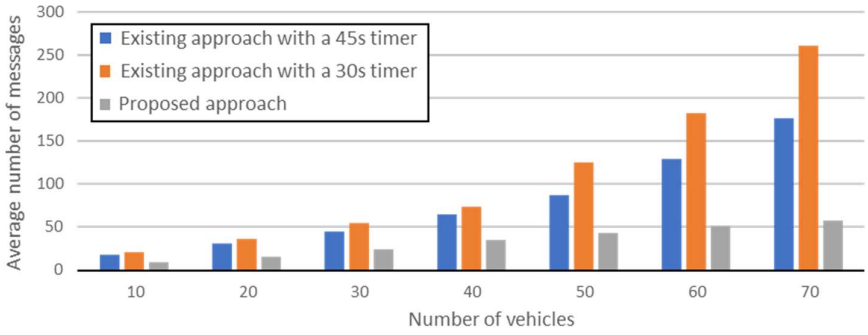


Fig. 4-39: Communication Overhead Comparison

It is clear from the Fig. 4-39 that the overhead is higher in Mühlbauer et al [14] with both timer durations than in the proposed framework. With a 30 second timer in approach Mühlbauer et al [14], the communication overhead is two, three, and four times higher than the proposed framework when the number of vehicles is 50, 60, and 70, respectively. In most situations, the overhead in the proposed framework is significantly lower than the baseline approach, which suffers from a higher overhead due to the need of generating feedback towards the RSU for regular reputation updates.

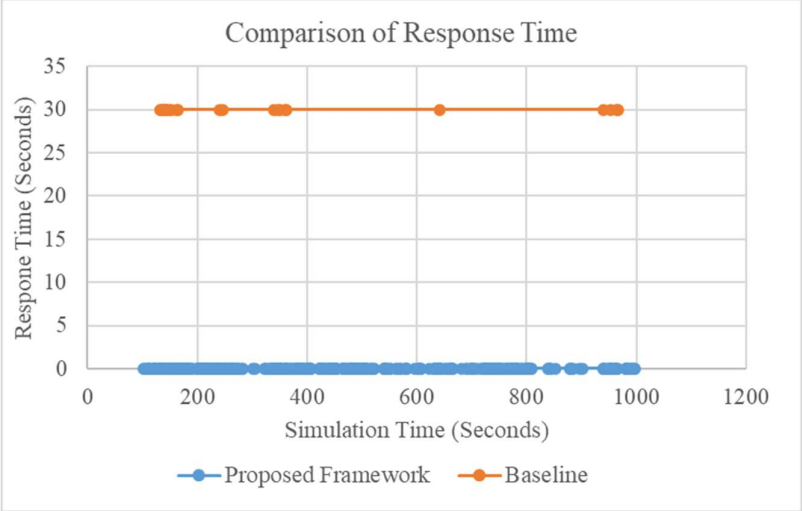


Fig. 4-40: Comparison of Response Time

In addition, as expected, the proposed framework is better when we compare its response time against a receiver-side evaluation-based trust approach. The baseline employs a timer for a predefined period i.e., 30 seconds, to collect additional messages about the same event. When it expires, receivers decide on an event. This is why it suffers from a higher response time which is shown in Fig. 4-40 where the red-coloured line signifies the constant response time for all receivers to decide about an event. During this time, vehicles may enter a "problematic" road area as they are typically moving fast. On the other hand, the proposed framework quickly decides an event without further communication unless it is disputed. This is why the response time of receiver vehicles is zero which can be considered

as the decision time for an event to drivers. This is calculated by deducting message arrival time from the current time which is zero seconds as indicated by the blue line in Fig. 4-40. Thus, the proposed framework exhibits a faster response time as compared to Mühlbauer et al [14].

4.7 Summary

In this chapter, the proposed trust model is validated using the Veins simulator. Untrue and inconsistent attacks are generated from the regular vehicles, then they are reported by reporters and detected by RSUs. After this, malicious vehicles are punished based on number of times they launch attacks. When malicious cases reach three from the same driver / vehicle or the trust of driver drops to 0.05, then the access of a driver to the network is blocked. We also consider the accuracy of the proposed trust model by varying the percentage of malicious and benevolent feedback from trusted clarifiers. Finally, the framework is compared with a baseline in terms of communication overhead and response time to demonstrate the better performance of the proposed trust scheme.

Chapter 5: Markov Chain Driver Behaviour Model

5.1 Introduction

The trust framework uses memory-less, fixed probability-based driver behaviour model which does not capture the change of behaviour of drivers with different trust states. Also, driver's behaviour is not examined using different trust scores with the trust framework. In truth, current behaviour of a driver is a consequence of past behaviour, and it is expected that the future behaviour would reflect from the current trust. The Markov-chain model captures this behaviour where a driver sending trustworthy messages builds trust and vice-versa. With this driver behaviour model, driver's announcement and untrue attack reporting behaviour are analysed. This model is flexible in number of states, so anyone can modify the state to meet the purpose.

This chapter presents a Markov chain–based driver behaviour model for the proposed trust model described in Chapter 3. There are six different trust states for the proposed Markov chain model and from each of these states, their lying probabilities are defined to examine their honesty or lying behaviour. These states are defined based on the different trust thresholds set for the framework. Trust states are ordered according to the increasing trust values. Thus, a driver who wants to reach a higher trust state must achieve a higher trust value. A driver switches to another state when its trust score falls outside the range of trust scores for the current state. A driver with a higher trust state has a higher probability to announce more trustworthy messages than those with a lower trust state. With this model, acceptable behaviour means announcing trustworthy messages whereas unacceptable behaviour means announcing untrue messages. When a trustworthy message is announced, a driver improves the trust score from it with the framework, but this reward is not given for the sake of the behaviour analysis. If another driver sends a report about it and the sender driver wins the dispute, then the RSU reward is added to the current trust. As a result, the sender driver possibly makes a transition to another state which is associated with higher trust scores than the current one. In contrast, a driver's trust score is reduced from the announcement of an untrue message which is proven to be malicious by an RSU. Whether an announcement is trustworthy or untrue, it is directly related to the behaviour of the real-world driver. Hence, these activities are simulated with the proposed Markov chain-based state transition diagram by setting the probabilistic distribution of controlled untrue and trustworthy message announcements from each state. A driver earns rewards from the forwarding, announcement, and gets either reward or punishment from RSU if there is a dispute relating to his/her announcement.

5.2 Existing Research on Markov Chain Model for VANETs

In Gazdar et al [169], the researchers present a Markov Chain-based hybrid trust model for VANETs. In this model, a Markovian state transition model, and the state transition probabilities are presented considering the cooperating factor in forwarding messages and the accurate evaluation of the received messages. The monitoring process considers trustworthy message broadcasting besides considering

cooperativeness. Predictions are made about staying in one state and reaching a particular state at some time. They have examined camouflaged behaviour with this Markovian model. It requires vehicles around the monitored vehicles to monitor the activities. Goli-Bidgoli et al [170] also present a Markov chain-based trust management model for Cognitive Radio (CR)-VANETs which derives the event occurrence probability using the Markov chain to keep the decision delay within an acceptable limit. Also, this model introduces a state transition model and probabilities to control the movement between the states based on the agree or disagree opinions. Haddadou et al [13] consider a distributed trust model based on a job signalling scheme which allows vehicles to determine the cost to access information. This cost is paid back when an announcement is trustworthy. This model also uses a Markov chain to validate the trust theoretically. Liu et al [171] propose a Hidden Markov Model (HMM) based trust evaluation method which computes trust of vehicles at the RSUs. This model improves the accuracy in detecting malicious vehicles than an existing model.

5.3 Proposed Markov Chain-Based Driver Behaviour Modelling

The proposed Markov model has six different trust states out of which one is the access-blocked state. A driver reaches this state when he / she is blacklisted. Other states are associated with different ranges of trust values. The six trust states are: “very high”, “high”, “normal”, “bad”, “very bad” and “access-blocked”. The probabilities of sending trustworthy and untrue messages from these states are set as shown in Table 5-1. Table 5-2 lists the untrue attack generation probability from reporter drivers or defines the behaviour of the reporter drivers. These values are selected such that drivers in higher trust states send less untrue messages and reports than in the lower trust states. In Table 5-1 and Table 5-2, probabilistic distributions for the “access-blocked” state are not defined as drivers cannot announce or report any messages from this state. In a real-world scenario, a driver can react differently at different times which can be modelled with a Markov chain-based driver behaviour model using a different probabilistic distribution. So, it is possible to examine their behaviour in different ways with a different probabilistic distribution which may result in different actions. Table 5-1 shows the lying probability of the sender driver which can be configured with different values to simulate the variation in driver behaviour. Similarly, the probability of sending untrue attacks from different trust states in Table 5-2 can be changed to model variation in reporter drivers’ behaviour.

Table 5-1: Driver’s Announcement Lying Probability

States	Probability of Announcing Trustworthy Message	Probability of Announcing Malicious Message
“Very good”	0.8	0.2
“Good”	0.6	0.4
“Normal”	0.4	0.6
“Bad”	0.2	0.8
“Very bad”	0.1	0.9

Table 5-2: Reporter's Untrue Attack Reporting Probability

Announcement from a sender with Trust State	Probability of Reporting an Untrue Attack	Probability of Not Reporting an Untrue Attack
"Very good"	0.1	0.9
"Good"	0.3	0.7
"Normal"	0.5	0.5
"Bad"	0.7	0.3
"Very bad"	0.9	0.1

With these trust states, a Markovian state transition-based driver behaviour model is presented, which is consistent with the trust framework described in Chapter 3. Also, this model is implemented separately for each driver of a vehicle. A diagram of this model is shown in Fig. 5-1. It has fixed trust states, and each state is associated with a range of trust scores. A driver stays in one state when his / her trust belongs to the range of trust values related to that state. This model also specifies the lying probability of a driver to control his / her behaviour. With this model, a driver starts his/her journey from the "normal" state with a trust value equal to 0.5. From this state, a driver sends some events and relays events from other vehicles to achieve a higher trust score. But this model only analyses the announcement lying behaviour from drivers. Thus, from a "normal" state, a driver can build trust to reach the "good" state if he / she continues announcing trustworthy messages in the network. Also, he / she can lose trust by announcing untrue messages to reach the "bad" state from the "normal" state. He / she can even move to the "very bad" trust state if most of the announcements are untrue. In the worst case, the driver may be access-blocked if his/her trust score reaches 0.05 by announcing several untrue messages. Alternatively, from the "good" state, a driver can improve trust to move into the "very good" state to become a highly trusted driver. Once a driver is in the "very good" trust state, it is harder to lose trust as in this model he/she only announces untrue messages with 0.2 probability. As such, the model captures the philosophy that good drivers tend to remain so, and vice versa unless they are encouraged to modify their behaviour. For consecutive untrue message announcements, the driver's trust score is reduced. In this case, he / she may be moved to the "good" or "normal" state. It is even possible to move into the "bad" or "very bad" state when he turns severely malicious. In this way, a mal-intent driver loses his/her trust and may be access-blocked in the network from where he / she cannot participate in any communication. When a vehicle is access-blocked, we assume an external procedure is followed to enable him/her to be reset to the "normal" trust state, if permitted.

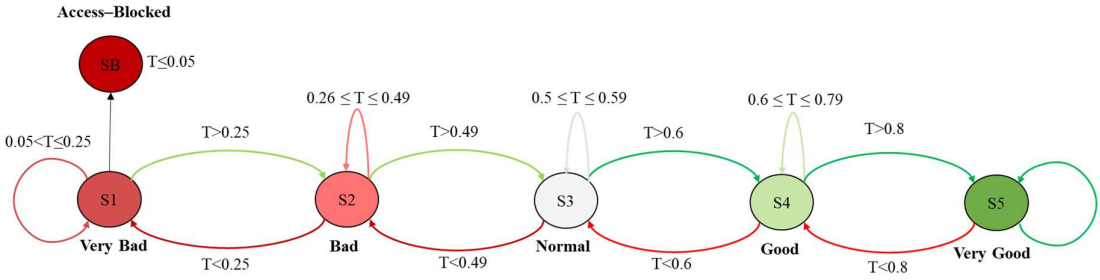


Fig. 5-1: Markov-Chain Behavioural Model (State Transition Diagram)

A clarifier is a vehicle which sends feedback in response to an RSU query. This feedback is kept consistent with the driver behaviour model. This allows the behaviour of clarifiers to be programmed similarly to the probabilities defined for different trust states of the sender and/or reporter drivers. As the trust model does not evaluate a clarifier's feedback, their behaviour analysis is not considered as important as the sender or reporter information.

5.4 Analysis and Validation of the Markov Chain Driver Behaviour Model

5.4.1 Experiment Setup

A set of experiments has been carried out to evaluate the behaviour of sender or reporter drivers by changing their lying probability to observe the proportion of trustworthy and untrue messages generated from different trust states over the simulation period. The participating vehicles run on a fixed circular route in the Erlangen city map during the simulation. Though a hundred vehicles participated, the behavioural analysis of only one sender driver and five reporter drivers is examined in this instance. The proposed trust model is implemented in Veins which comprises OMNeT++ and SUMO. The series of experiments run for 5000 simulation seconds and then the trust scores of the drivers that are involved in conflicts via the RSU are measured. These vehicles experience a warm-up period without announcing any event. The vehicle numbers are maintained constant throughout the simulation. When the warm-up period is elapsed, a fixed sender driver announces messages periodically at 1000s intervals starting at 500s. Multiple types of events are announced from the same driver for analysis. The announcements are scheduled as an accident message at 500s, a debris message at 700s, a road defect message at 900s, a traffic element problem at 1100s and a tree on road message at 1300s. Reporters deterministically send untrue attack reports based on the probabilistic distribution defined in Table 5-2. The drivers of vehicles V[1], V[2], V[3], V[4], and V[5] are the only reporters of events announced by the driver of vehicle V[0] for the series of experiments conducted. As we wish to model the behavioural change of these reporters as well, their trusts are shown on the charts beside the sender driver. In this way, a series of experiments are conducted with different initial trust distributions and then the trust evolution is observed to examine the distinctive driver behaviour.

A fixed reward and punishment is used from the disputes to update the trust of drivers since only from the disputed data we can differentiate their behaviour, whether they lie or not and what condition they lie mostly. Other rewards and punishments within the trust framework are not enabled for this analysis of driver behaviour. For example, rewards and punishments from announcements as well as forwarding and beaconing rewards are omitted to aid the visualization of RSU rewards and punishments. In this series of experiments, drivers can send untrue attacks even when their trust score is less than 0.5 which was not allowed with the trust model presented in Chapter 3. If a driver can send a message from a particular trust state, then he/she is allowed to send an untrue attack version of the originated message. For example, if an event announcement needs at least a trust score of 0.26, then a reporter can send an untrue attack with this trust score. An RSU employs a 120-second timer to determine the validity of a dispute from the clarifier feedback. Thus, the verification time delays the reception of rewards and punishments from an RSU. Also, two different types of messages are used to disseminate the RSU reward or punishment to the drivers which also adds an additional delay besides their availability to an RSU and wireless collisions.

There are two sets of experiments conducted for examining the driver behaviour model. In the first set of experiments, clarifiers send opinions based on the witness of the event and a probability distribution. If a driver with a "very high" trust state generates an event, then the clarifiers send positive opinions with 0.8 probability and negative opinions with 0.2 probability. For the "good" trust state, clarifiers send positive opinions for 60% of cases and negative opinions for 40% of cases. A message from a "normal" trust state originating from a driver results in 40% positive and 60% negative opinions. From the "bad" state, clarifiers deny announcements 80% of the time and support only 20% of the time. From the "very bad" state, clarifiers deny announcements 90% of the time and support only 10% of the time. This distribution can be changed as needed to model various sender or reporter driver behaviour. In the second set of experiments, clarifiers send feedback based on the probability distribution of their trust states as shown in Table 5-3 and the reporters send reports based on Table 5-2.

5.4.2 Scenarios of Behavioural Analysis of the Drivers

5.4.2.1 Uniform Trust Distribution (0.4 to 0.5)

In this experiment, initially, all vehicles are inserted, and drivers are assigned their initial trust using a uniform distribution in the range of 0.4 to 0.5. Fig. 5-2 records the lying behaviour data from this experiment. The x-axis shows the simulation seconds, and the y-axis shows how trust score changes from the rewards and punishments. There is an accident message scheduled from V0 which is not announced as the trust of the driver is not sufficient. This is why a change in trust data commences from 700s when the driver of V[0] announces a debris message. As the trust of the driver of V[0] is low, he/she has a higher chance to lie to others which is modelled using a probabilistic distribution. As the driver of V[0] lies, the drivers of V[2] and V[3] improve their trust by sending untrue attacks and they

win against the driver of V[0]. This is visible from the chart. The other two drivers do not participate in the reporting process and hence their trust remains constant over the simulation period. Also, V[5] wins one dispute over V[0] which is indicated by a trust increment in its characteristic at about 3600s. Though 100 drivers have participated, the trust records of other drivers are not included in this chart for simplicity as their trust remains constant.

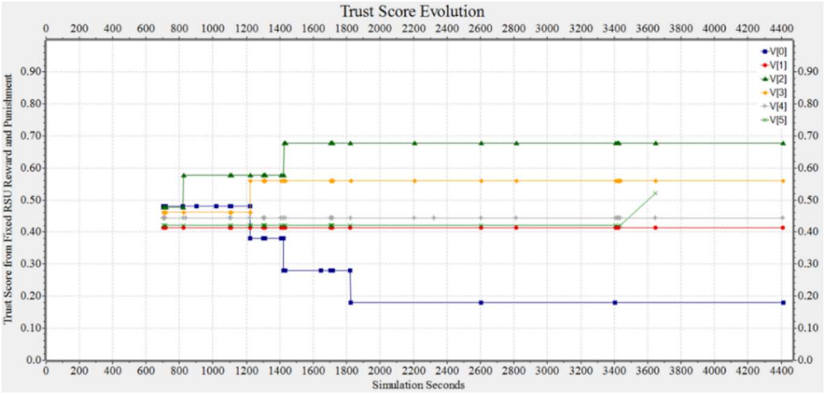


Fig. 5-2: Behavioural Analysis of the Drivers of V[0], V[1],…,V[5] with Trust (0.4-0.5)

5.4.2.2 Uniform Trust Distribution (0.5 to 0.6)

In this experiment, regular vehicles are assigned the initial trust in the range of 0.5 to 0.6 using a uniform distribution. Fig. 5-3 records the lying behaviour data from this experiment. The x-axis shows the simulation seconds, and the y-axis shows trust score from the rewards and punishments only. When the first message is announced, it is reported which can be identified by the trust decrement of V[0] near 620s. Though it is random behaviour, trust can go upward but unfortunately, the trust of the driver of V[0] shows a downward trend and it is access-blocked at about 3200s. As the driver of V[0] loses all disputes, any reporter driver who sends untrue attacks improves trust from RSU rewards. This trend is noticeable for the drivers of V[1], V[2], and V[5] where the driver of V[2] improves trust score greatly to about 0.8 by the end of the simulation. We can attribute this to V[0] lying repeatedly and the reporters were trustworthy throughout the simulation. The trust of the other two drivers of V[3] and V[4] remained constant throughout the simulation as they did not send any untrue attacks during the simulation period.

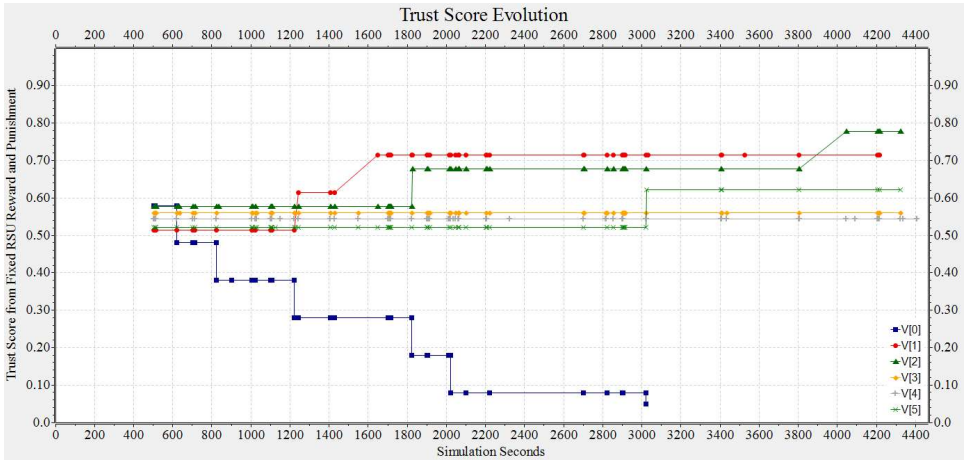


Fig. 5-3: Behavioural Analysis of the Drivers of V[0], V[1],…,V[5] with Trust (0.5-0.6)

5.4.2.3 Uniform Trust Distribution (0.6 to 0.7)

This experiment is conducted under the same conditions as Section 5.4.2.1 and 5.4.2.2 except the initial trust values of the drivers are allocated from the range 0.6 to 0.7 using a uniform distribution. Fig. 5-4 records the lying behaviour data from this experiment. In this experiment, though the driver of V[0] has an initial trust score of 0.7, he/she always originates trustworthy messages that result in the trust score improvement from the RSU rewards. This is an instance demonstrating acceptable behaviour from a sender driver like V[0]. There is no trust score change noticed for the driver of V[0] after the 2800s as the maximum trust is reached. The driver of V[4] does not participate in any conflict, so his/her trust remains constant. Other reporter drivers V[1], V[2], and V[3] receive RSU punishments when they send reports which reduce their trust as the simulation time progresses. The trustworthy driver finishes the simulation with a higher trust score of 0.9 and the malicious reporters finish with a lower trust score than their initial value.

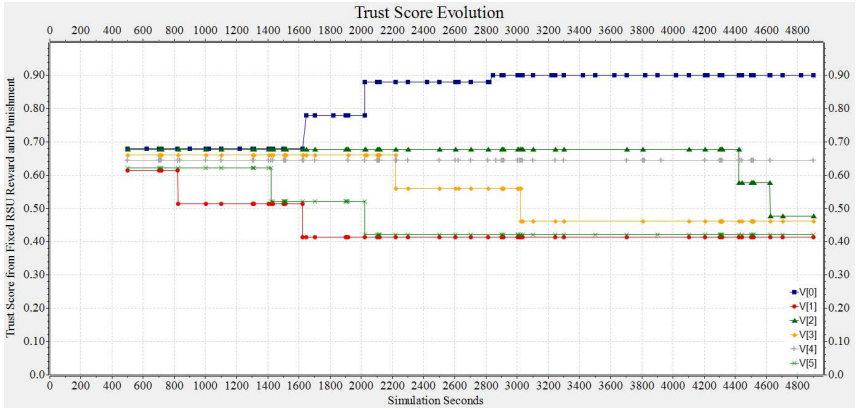


Fig. 5-4: Behavioural Analysis of the Drivers of V[0], V[1],…,V[5] with Trust (0.6-0.7)

5.4.2.4 Uniform Trust Distribution (0.7 to 0.8)

In this experiment, initial trust values are set between 0.7 and 0.8 from a uniform trust distribution. Fig. 5-5 records the lying behaviour data from this experiment. After the message announcement begins no trust changes are noticed until the 1300 seconds as there is no untrue attack reported from the reporters. As the driver of V[0] always sends trustworthy messages, it receives RSU rewards, and its trust characteristic has an upward trend after 2000 seconds. The trust of the driver of V[0] only increments as he/she sends only trustworthy messages even though some malicious reporters with good trust scores challenge this message. As these reports are malicious, they receive RSU punishments whereas the driver of V[0] receives RSU rewards. Drivers of vehicles V[4] and V[5] do not send any report as V[0] sends only trustworthy messages so their trust characteristics remain unchanged throughout the simulation.

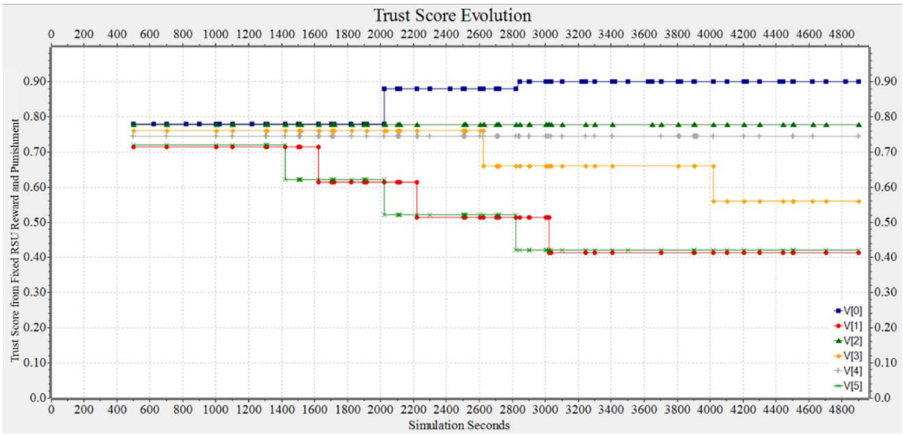


Fig. 5-5: Behavioural Analysis of the Drivers of V[0], V[1],…,V[5] with Trust (0.7-0.8)

5.4.2.5 Uniform Trust Distribution (0.8 to 0.9)

During this experiment, drivers are assigned the "very high" trust score in the range of 0.8 and 0.9. Fig. 5-6 records the lying behaviour data from this experiment. From the results, there are no trust score changes seen until 1400 seconds as no reporter sends any untrue attack and they are also assigned a "high" trust state. Suddenly, the driver of V[2] sends an untrue attack as this process is random. Because of this, the driver of V[0] receives a reward and the driver of V[2] is punished. Overall, fewer attacks are reported than in the previous experiments as the reporters are also assigned a higher trust state than in the former experiments. The driver of V[0] builds trust throughout the simulation.

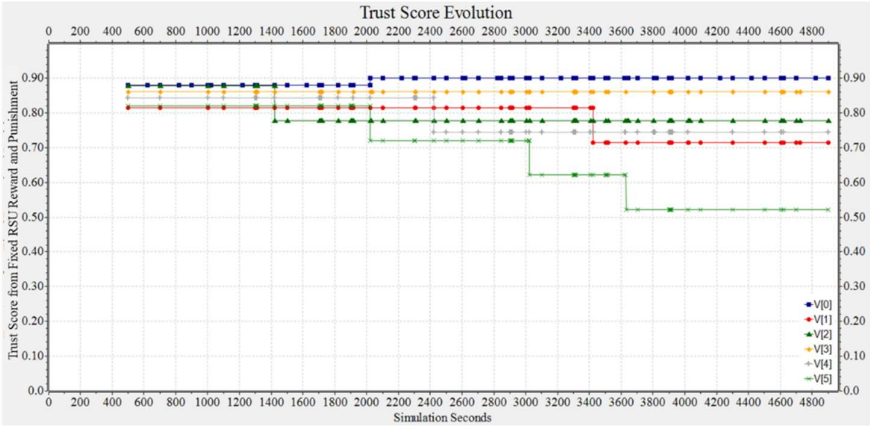


Fig. 5-6: Behavioural Analysis of the Drivers of V[0], V[1],…,V[5] with Trust (0.8-0.9)

5.4.2.6 Fixed Trust Score of 0.9

In this experiment, all drivers start from a very high trust state with a trust score of 0.9. Fig. 5-7 records the lying behaviour data from this experiment. The driver of V[0] sends 90% trustworthy and 10% of malicious announcements from this state. It is seen very few announcements are reported from V[1] and V[5] as they are also assigned "very good" trust states though their malicious probability is 0.1. This results in the constant trust score of the driver of V[0] while some reporters send untrue attacks maliciously which are disproved at RSUs. Hence, some reporters receive RSU punishments at different times during the latter part of the simulation. The drivers of V[1] and V[2] send only untrue attacks for which their trust is reduced.

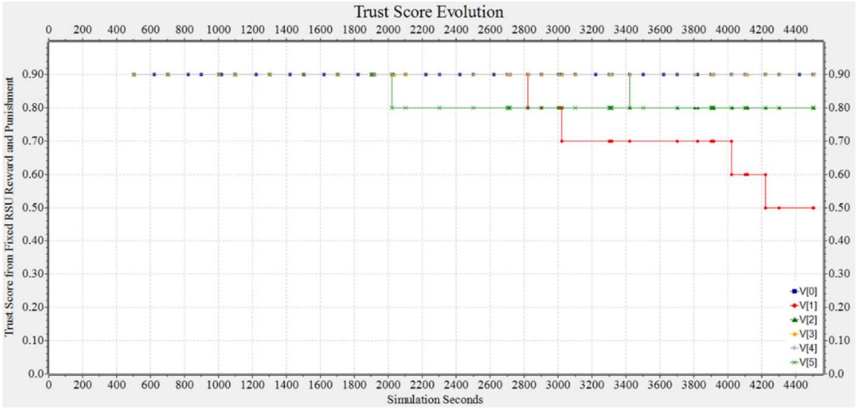


Fig. 5-7: Behavioural Analysis of the Drivers of V[0], V[1],…,V[5] with Trust=0.9

5.4.3 Discussion on Results

The drivers are assigned different trust states and scores in different experiments. Their lying characteristics are controlled using a probabilistic distribution. The announcement of trustworthy messages varies based on the driver's trust state. With a higher trust state, there are less untrusted messages announced, as configured. The number of untrusted messages rises for the lower trust states. The reporter drivers send less untrue attacks and more accurate reports associated with their trust scores

in the higher trust states. The simulation of this model returns expected results and the driver's untrue message generation and the reporter's reports are controlled based on their current trust states. It is possible to modify the probability distribution to simulate different behaviour of the sender and reporter drivers.

5.5 Behavioural Analysis of Sender with Fixed Trust (0.6) of Reporter and Clarifier

5.5.1 Experimental Setup

The set of parameters are same as those for the set of experiments conducted in Section 5.4. 100 vehicles are added and then they experience a warm-up period. One sender driver, V[0], sends messages periodically and five reporters from V[0],…,V[5] send reports based on the probability distribution defined in Table 5-2. Table 5-3 lists the feedback generation probability of clarifier vehicles. In the next two experiments, clarifiers send feedback based on the probability distribution of their trust states. Also, the reporters send reports based on the probability distribution of their trust states After this their behaviour is captured in Fig. 5-8 and Fig. 5-9.

Table 5-3: Clarifier Feedback Distribution

Trust States	Probability of Sending Positive Feedback	Probability of Sending Negative Feedback
"very good"	0.8	0.2
"good"	0.6	0.4
"normal"	0.4	0.6
"bad"	0.2	0.8
"very bad"	0.1	0.9

5.5.2 With Sender Driver Trust of 0.3

Fig. 5-8 shows the trust score evolution of six vehicles. In this experiment, the trust of the sender driver is set to 0.3, and the trust of the reporter and the clarifier are set to 0.6. Clarifiers send opinions when an RSU asks based on their probability distribution of trust states. As the trust score of the reporter and clarifier belong to the "Good" trust state, from this state reporters send untrue attacks with 0.3 probability and do not send untrue attacks with 0.7 probability. Clarifiers with a "Good" trust state send positive feedback with a 0.6 probability when they visit the event location. They send negative feedback with a 0.4 probability if they do not see the event at the said location. Until first 1400 seconds, there is no dispute, and no trust change is observed. After this, there are many reports announced for which the driver of V[0] wins as the reporters send malicious reports. The reporter driver of V[5] loses all disputes which reduces their trust to 0.2 at 2400s. The driver of V[3] does not report any announcements from V[0] until 2800 seconds as seen from the chart. After this time, V[3] sends many reports to the RSU which are proved false, so its trust is reduced to 0.2 at 4030 seconds. Other reporters excluding V[4]

occasionally send untrue attacks and lose disputes to V[0]. In this way, V[0] builds trust as it always announces trustworthy messages and some reporters, being malicious, lose trust.

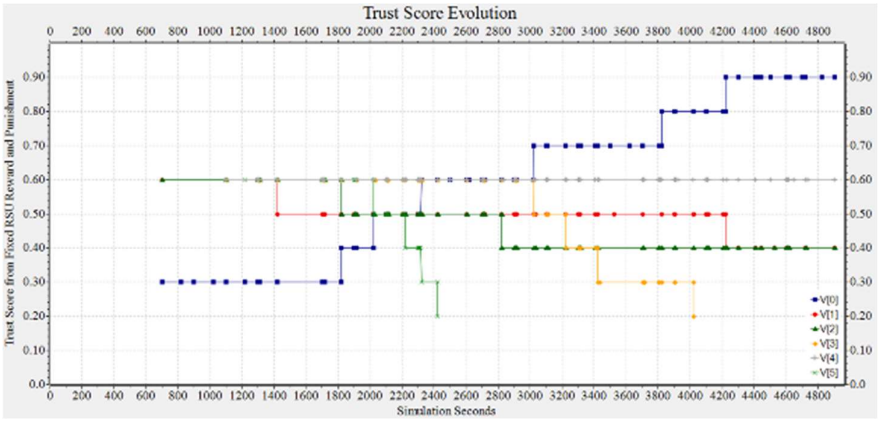


Fig. 5-8: Behaviour Analysis of Driver When Trust Score is 0.3

5.5.3 With Sender Driver Trust of 0.7

In this experiment, the sender driver starts with 0.7 trust in the "good" trust state whereas the clarifiers and reporters have a trust state that is the same as the previous experiment. In this experiment, the driver of V[0] only builds trust as it always sends trustworthy announcements. Reporter drivers from V[2], V[3], and V[5] send reports maliciously for which they lose all disputes. These are noticed by the trust decrements in Fig. 5-9. Reporter V[4] does not send any report and V[1] sends only one untrue report for which it receives an RSU punishment. When a reporter sends a malicious report and receives RSU punishment, it subsequently sends more reports maliciously as its trust state moves to "bad", as expected.

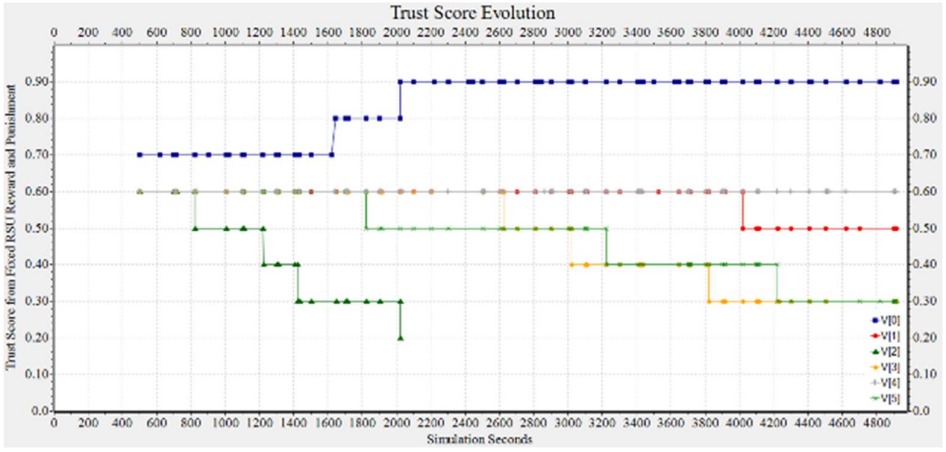


Fig. 5-9: Behaviour Analysis of Driver When Trust Score is 0.7

5.5.4 Discussion on Results

The sender driver of V[0] starts with a trust of 0.3 and 0.7 in two experiments whereas, the trust of the reporters and clarifiers is unchanged which is 0.6. They both start in the "Good" trust state. With

these settings, reporter vehicles send untrue attacks in 30% of cases and clarifiers send positive opinions in 60% of cases when they observe an event on the road. It is seen in both experiments that the sender driver only improves his / her trust in spite of some reports which are proved false by RSUs. The sender remains trustworthy throughout the simulation and reporters receive RSU punishment which reduces their trust, and they move to “normal”, “bad”, and then “very bad” trust states in consequence. The reporters send more false reports from the normal and bad states which reduces their trust score. The sender driver reaches the “Good” trust state early in Fig. 5-9, it remains so and vice versa.

5.6 Summary

In this chapter, a Markov-chain based driver behaviour model is presented for VANETs. We have analysed driver behaviour in different conditions and then measured their trust from the RSU rewards and punishments. The results confirm both the sender and reporter drivers with the higher trust score launch fewer untrue attacks than the drivers with a lower trust state.

Chapter 6: Fuzzy Logic-Based RSU Reward or Punishment Assessment Scheme

6.1 Introduction

In the current framework, fixed RSU rules are used to assess the validity of a disputed event using the feedback from trusted clarifiers. To decide if a driver/vehicle has sent an untrue attack, the RSU computes a sum of products of trust and feedback and then checks whether the result is greater or less than zero to determine the benevolent or malicious drivers. Then the RSU issues a fixed amount of reward or punishment to the respective drivers without considering the incident type, the driver's past behaviour, and the nature of the supporting feedback. Thus, this research now extends the trust framework by incorporating these factors in the evaluation of the reward or punishment for the concerned drivers. However, uncertainty exists about these parameters. Guleng et al [15] and Soleymani et al [130] use fuzzy logic to handle uncertainty and inexactness within VANETs. For example, in Guleng et al [15], fuzzy logic is used to calculate the direct trust of neighbour vehicles from cooperativeness, honesty, and responsibility factor. Furthermore, fuzzy logic can mimic human decision-making capabilities. Hence, we apply fuzzy logic to determine the reward or punishment for conflicting drivers from the severity of incident, driver past behaviour, and the feedback nature.

6.2 Existing Fuzzy Logic Applications for Trust Management

Agrawal et al [172] propose a Fuzzy Logic Based Greedy Routing (FLGR) protocol which selects the best relayer using fuzzy logic. The next hop is selected from the current node using the maximum distance, speed, and angular deviation. This approach only considers the current state of a vehicle when selecting the next forwarder vehicle. Zhou et al [173] select an optimal path for packet forwarding using a fuzzy logic-based transmission method. In this method, driving direction, vehicle speed, link time, and hop count are used for relay node selection. It is better than Agrawal et al [172], as it considers the future state of vehicles as well. In Igried et al [174], a fuzzy logic-based trust model is proposed that uses the RSU assessment, emulation attack attempt, and collaboration degree to assess the trust of vehicles. It incentivises good behaviour and punishes malicious vehicles. However, their analysis only concentrates on network performance considering the malicious behaviour of slowing connections, modifying messages, and stating false opinions. There is no attack detection scheme.

Inedjaren et al [175] extend the Optimized Link State Routing (OLSR) protocol with fuzzy trust. In this method, vehicles periodically send control messages (Hello and TC messages) in the locality to evaluate the trust using fuzzy logic. This model can avoid blackhole attacks. In Hasan et al [176], a fuzzy logic-based trust model is proposed to address uncertainty and inaccurate trust estimation. In this method, edge servers compute the trust of vehicles using fuzzy logic from packet drop, alteration, and

false message injection factors. The analysis considers message alteration attacks and bad-mouthing attacks. In Gayathri et al [177] a Mamdani fuzzy inference-based fuzzy logic system is used for vehicle authentication. This system only considers distance and trust factors to classify vehicles as partially or fully trusted, or malicious. This approach is not analysed in the presence of a known adversary model.

Xia et al [178] propose a fuzzy logic-based multicast routing protocol considering node and path trust. In this approach, fuzzy logic is used to combine the direct and indirect trust into the final trust. With this score, malicious vehicles are removed to maintain a trusted path though end-to-end delay and control overhead increase slightly. In Soleymani et al [179], a fuzzy-logic-based trust model is presented where plausibility, experience, and vehicle type are used to decide on the validity of events. The fuzzy decision-making module of receiver vehicles utilizes these factors to compute the trust of the sender before accepting or rejecting or forwarding messages. The analysis considers simple (i.e. it prevents obtaining services via other nodes), opinion tampering, and on-off attacks. Every receiver vehicle applies fuzzy logic independently before forwarding traffic messages to further vehicles. Malhi et al [180] propose a fuzzy system considering network density, relaying distance, and trust inconsistency to predict the relaying trust of vehicles. Then coordinated trust is computed using velocity, connection degree and connection loss parameters. After this, the final trust is computed using a fuzzy system considering the relaying and coordinated trust to then select a trusted path.

6.3 Fuzzy Reward / Punishment Parameter Selection

An RSU should allocate a higher reward when a driver's recent history only consists of good behaviour whereas he / she should receive a lower reward when his/her most recent history contains bad behaviour. In contrast, a driver should receive more punishment if his/her history is bad because he has not shown trustworthy behaviour in the recent past. However, the punishment can be lower if the recent history shows negligible punishing actions compared to rewarding actions. As these data are stored at the TA, an RSU can easily retrieve them and perform a fuzzy reward and punishment assessment. The TA maintains a driver profile database to store a maximum of ten reward or punishment records for every driver which are collected from the previous dispute decisions. A limited number of records are kept as a driver could be involved in many disputes over time which makes this size too large. If the system is configured to store higher number of records for each driver, then the TA needs to allocate more storage whereas a small number of records would not be sufficient to characterize a driver's past behaviour. Therefore, it is required to store a moderately sized driver past behaviour record size which is set to ten for each driver by default. If a driver is always trustworthy, then he/she would receive more rewards from the conflicts. If a driver generally becomes bad, he receives a lower reward from the assessment. However, this is not only the determining factor to set the level of punishment or reward as it also relates to the severity of incident and RSU confidence.

An RSU confidence score from the received feedback is estimated from the extent of support or opposition to the original sender or the reporter. The fixed RSU reward / punishment evaluation scheme does not analyse the received feedback, i.e. whether they are mostly supportive or opposing to a driver announcement. The supporting / opposing nature of the feedback is normalized and it is called the RSU confidence in the sender or reporter. RSU confidence in the sender is the ratio of supporting reports to the total reports. In contrast, an RSU confidence in the reporter is the ratio of opposing reports to the total reports (support and opposing).

If SF is the supporting feedback, OF be the opposing feedback, RSU-CS be the RSU confidence in the sender, and RSU-CR be the RSU confidence in the reporter, then Eqn (6-1) and Eqn (6-2) state the RSU confidence score in the sender or the reporter, respectively.

$$\text{RSU-CS}=\text{SF}/(\text{SF}+\text{OF}) \tag{6-1}$$

$$\text{RSU-CR}=\text{OF}/(\text{SF}+\text{OF}) \tag{6-2}$$

The numerator is selected by the way sender or reporter announces the event. For example, the sender says there is an event whereas the reporter says an opposite event relating to the sender's event. When the reward or punishment is assessed, if the RSU-CS is higher, then the reward is possibly higher if the sender wins the dispute. In contrast, If the RSU-CR is higher, then the punishment is possibly higher when the reporter loses the dispute. However, the amount of reward and punishment also depends on two other factors. The trust model assigns reward or punishment without considering the severity of the event. Different types of traffic events have different levels of detrimental effect on human lives. Hence, the reward or punishment for announcing an untrue accident event or a debris event should not be the same.

A driver's past behaviour, severity of an incident, and RSU confidence in the sender or reporter while resolving a dispute with the current model provides measures that allow for more nuanced assignment of rewards and punishments. We therefore propose a fuzzy logic-based RSU controller that accounts for these three factors when assigning a reward or punishment for a driver. Fuzzy logic adequately handles the imprecision when deciding on an approximately reasonable amount of reward and punishments from the RSU when disputes arise.

6.4 Overview of the Proposed Fuzzy RSU Assessment Scheme

Fig. 6-1 depicts the fuzzy RSU assessment scheme for reward or punishment. It starts from the left-hand side through which it collects three inputs which are driver past behaviour, confidence in the sender or reporter, and severity of an incident. This requires processing of input data to insert them into the fuzzy application. Then these inputs are handed over to the fuzzifier to produce input fuzzy sets. These sets are delivered to the fuzzy inference module which evaluates the fuzzy rules to produce output fuzzy sets. These sets are then transferred to the defuzzifier module to generate the crisp number as an

output which is sent to the respective drivers as reward or punishment. This system provides reward or punishment as output variables. A dispute decision at an RSU invokes the execution of the Fuzzy logic-based reward or punishment application to determine the extent of reward or punishment for a conflicted announcement.

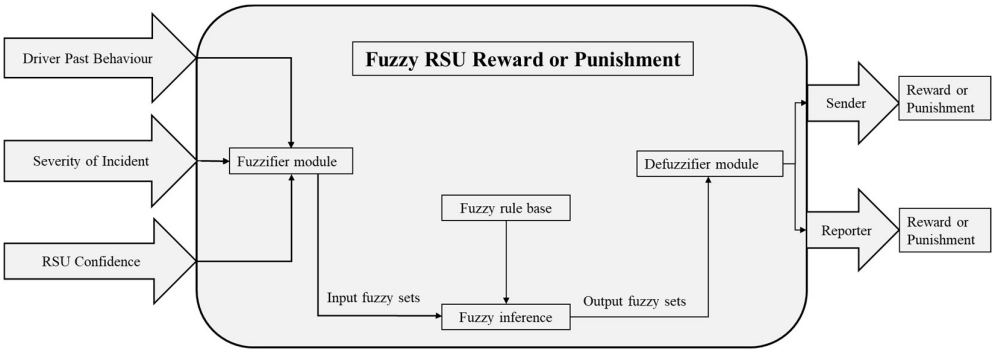


Fig. 6-1: A Block Diagram Representation of the Proposed Fuzzy RSU Assessment

6.4.1 Fuzzification

In this step, the system collects the three inputs and finds their degree of membership to the fuzzy sets using membership functions. First, the shape of the membership function for each input is defined intuitively. Then the degree of belonging to the fuzzy sets are determined for each input. Membership functions are defined with the help of linguistic variables which are shown in Table 6-1. The fuzzification of the input parameters map to the fuzzy values which is the degree of belonging to one or more linguistic variables (fuzzy sets).

Table 6-1: Input Fuzzy Sets

Input Parameters	Fuzzy sets
Driver Past Behaviour (DPB)	Good (G), Neutral (N), and Bad (B)
Severity of Incident (SI)	Not Severe (NS), Less Severe (LS), and High Severe (HS)
RSU Confidence Score (RCS)	Low (L), Medium (M), and High (H)
Reward (R)/Punishment (P)	Very Low (VL), Low (L), Medium (M), High (H), and Very High (VH)

6.4.1.1 Driver Past Behaviour

An RSU uses a membership function to convert each input to the degree of belonging to the fuzzy sets. RSUs always send data on the rewarded and punished drivers to the TA. An RSU asks for DPB data from the TA as it saves this information in a driver profile database. Let, NoP and NoR be the recorded number of rewards and punishments for the concerned drivers from their previous disputed events. When the TA sends NoP and NoR data to the dispute resolving RSU, then it estimates the ratio of NoP/(NoR+ NoP) for both drivers (When an official vehicle reports, then the RSU computes this

only for the sender driver and in many cases RSU may receive multiple complaints from multiple reporters about the same announcement in which case RSU ignores the repeated complaints). For each driver, the RSU feeds the data into the fuzzifier to obtain the degree of belonging for the DPB from the set: {"Good", "Medium", "Bad"}. Three fuzzy sets are used for the fuzzification of DPB. DPB ranges from 0 to 1 and each DPB is separated by 0.1 which is shown on the x-axis in Fig. 6-2. The y-axis shows the degree of membership. This figure illustrates the fuzzification of DPB. For DPB, it shows how to obtain the corresponding fuzzy sets. If we place a straight line parallel to the y-axis on a DPB level, then this line crosses with two fuzzy sets. The points where this line crosses are the corresponding degree of membership for the input DPB. When DPB is 0 or 0.1, then the **Good** fuzzy set is selected with a membership degree equal to 1. When DPB is between 0.2 to 0.4, **Good** and **Neutral** fuzzy sets are selected with a membership degree equal to the point at which the line crosses the two fuzzy sets. When DPB is 0.5, only the **Neutral** fuzzy set returns a membership degree of 1. When DPB is between 0.6 and 0.8, **Good** and **Bad** fuzzy sets are selected with membership degree equal to the point at which the line crosses the two fuzzy sets. When DPB is 0.9 or 1, only the **Bad** fuzzy set returns with a membership degree of 1. For example, if a driver record contains 4 punishments out of the 10 most recent records, then the DPB is 0.4. When the DPB is 0.4, the fuzzification returns the fuzzy value as {**Good**: 0.24, **Medium**: 0.76, **Bad**:0} since the line at DPB value of 0.4 crosses **Good** at 0.24 and **Neutral** at 0.76. If the degree of memberships is denoted by μ, then for a DPB of 0.4, **Good** and **Neutral** fuzzy sets are selected with the $\mu_{(DPB=neutral)}$=0.74 and $\mu_{(DPB=good)}$=0.26.

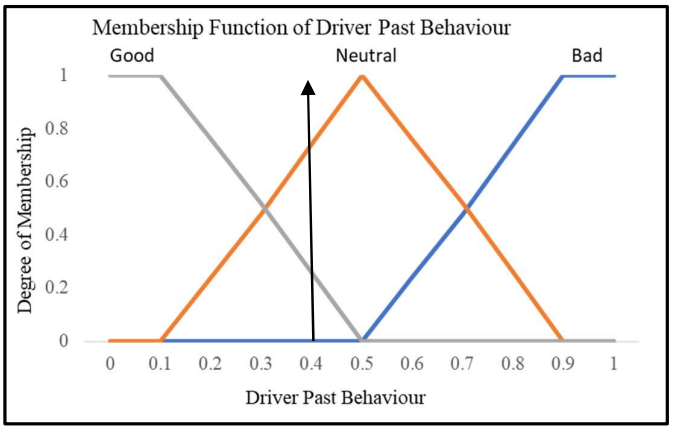


Fig. 6-2: Membership Function for Driver Past Behaviour

6.4.1.2 Severity of Incident

Each RSU stores a list of potential events with increasing severity level as shown in Table 6-2. These are not a complete list of possible events but provide an example. Other events could be added later. Also, events in this list are not based on an agreed standard. We provide an ordered list of events where each one has an increasing severity level from top to bottom in Table 6-2 based on the assumed impact on human lives. In this list, the event's name, and their corresponding severity level (assumed impact

on human lives) are shown. In the table the top row contains the least severe event, and the bottom contains the most severe type.

Table 6-2: Possible Event List

Incident Name	Severity Level (Lowest to Highest)
Road Clear	0
Debris or Road Spillage (Oil or Muds or Sands)	1
Illegal Waste Dumping	2
Poor Conditioned Road	3
Road Defect (i.e. Faded Sign,) or Malfunction Traffic Element	4
Stranded or Abandoned Vehicle or Obstacle or No Obstacle	5
Road Defect (Pothole)	6
Diversion or Road Maintenance	7
Severe Weather or Environmental Incident	8
Flood or Fallen Tree on Road	9
Congestion	10
Traffic Jam	11
Accident	12

Every RSU stores a copy of this table of potential events. When there is a dispute, an RSU looks up the severity level of an incident to feed it into the fuzzifier. Three fuzzy sets "Not Severe", "Less Severe", and "High Severe" are considered for this input. Next, the respective degree of membership for an input to the fuzzy sets is determined. Fig. 6-3 shows the membership function for the Severity of Incident (SI), where SI is shown on the x-axis for only 13 types of events from Table 6-2. The y-axis shows the degree of membership for the input value. This figure illustrates the fuzzification of SI to get the corresponding fuzzy sets. If we place a straight line parallel to the y-axis on an SI level, then this line intersects with two fuzzy sets. The points where this line crosses are the corresponding degree of membership for the input SI. When SI has a value from {0, 1, 2, 3, 4, 5}, then **Not Severe** and **Less Severe** fuzzy sets are selected with membership degree equal the point at which the line crosses the two fuzzy sets. When SI is 6, then only **Less Severe** is select with degree of membership equal to 1. When SI has a value from {7, 8, 9, 10, 11, 12}, **Less Severe** and **High Severe** are selected with membership degree equal to the point at which the line crosses the two fuzzy sets. For example, when the SI is 5, the fuzzification returns the degree of membership to the fuzzy sets {**Not Severe**: 0.18, **Less Severe**: 0.82, **High Severe**: 0} since the line on a SI value of 5 intersects **Not Severe** at 0.18 and **Less Severe** at 0.82. If the degree of membership is μ, then for an SI of 4, **No Severe** and **Low Severe** fuzzy sets are selected as $\mu_{(SI=not\ severe)}$=0.35 and $\mu_{(SI=less\ severe)}$=0.65.

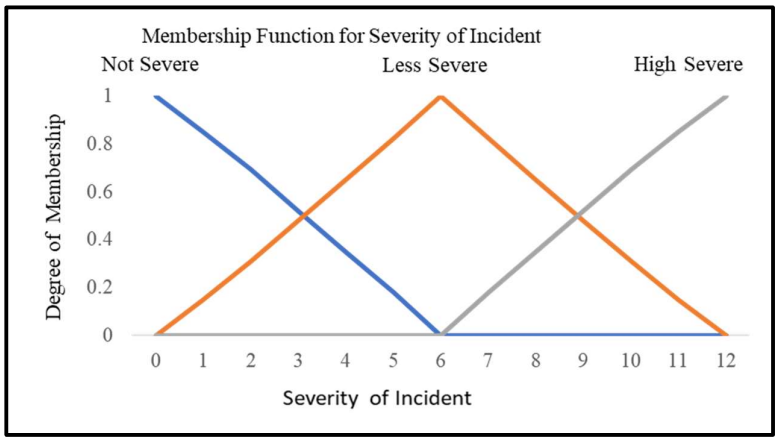


Fig. 6-3: Membership Function for Severity of Incident

6.4.1.3 RSU Confidence in the Sender or Reporter

An RSU calculates a confidence score in the sender from the received feedback as the ratio of feedback that supports the sender's announcement to the sum of the feedback which either supports or contradicts the announcement. Similarly, the RSU confidence in the reporter is defined as the ratio of feedback that supports the reporter's report to the sum of the feedback which supports and contradicts the reporter's report. We define three fuzzy sets for the RSU confidence which are "Low", "Medium", and "High". The corresponding membership function for RSU confidence is shown in Fig. 6-4.

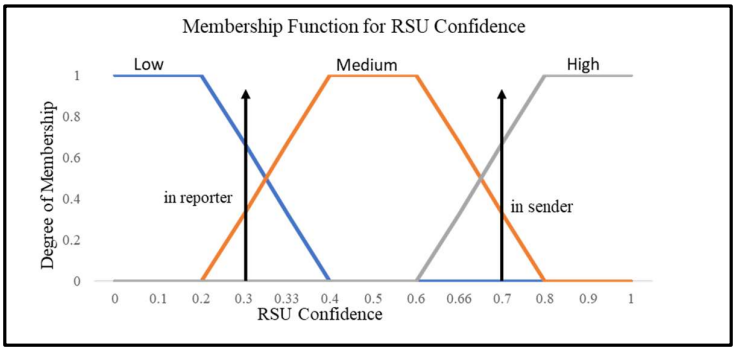


Fig. 6-4: Membership Function for RSU Confidence in the Sender or Reporter

In this chart, the x-axis shows the RSU Confidence (RCS), and the y-axis shows the degree of membership. There are two vertical lines in this chart which show that the confidence in the sender or reporter differ; in some case, these values may be the same. The degree of membership of the RSU confidence is the degree of belonging to the fuzzy sets "Low", "Medium", and "High". This figure illustrates the fuzzification of RCS to get corresponding fuzzy sets. If we place a straight line parallel to the y-axis on an RCS level, then this line crosses two fuzzy sets. The points where this line crosses are the corresponding degree of membership for the input RCS. When RCS has a value from {0, 0.1, 0.2}, then only the **Low** fuzzy set is selected with a membership degree equal to 1. When RCS has a

value from {0.3, 0.33}, then only **Low** and **Medium** fuzzy sets are selected with membership degree equal to the point at which the line crosses the two fuzzy sets. When RCS has a value from {0.4, 0.5, 0.6}, then only the **Medium** fuzzy set is selected with membership degree equal to 1. When RCS has a value from {0.6, 0.66, 0.7}, then only **Medium** and **High** fuzzy sets are selected with membership degree equal to the point at which the line crosses the two fuzzy sets. When RCS has a value from {0.8, 0.9, 1}, then only the **High** fuzzy set is selected with a membership degree equal to 1. For an RCS score of 0.7, the fuzzification returns as {**Low**: 0, **Medium**: 0.33, **High**:0.67} since a line on an RCS value of 5 crosses **Medium** at 0.33 and **High** at 0.67. If the degree of membership is μ, then for the RCS of 0.7, the degree of membership are $\mu_{(RC=medium)}$= 0.33 and $\mu_{(RC=high)}$= 0.67.

6.4.2 Fuzzy Rules for Reward and Punishment

A different set of rules are applied for reward and punishment evaluation. Table 6-3 shows the set of rules used for reward assessment whereas Table 6-4 is used for punishment assessment. The reason behind keeping two sets of rules is to vary the reward and punishment for a combination of three inputs. Let, Driver Past Behaviour be DPB, Severity of Incident be SI, RSU Confidence Score be RC, Reward be R, and Punishment be P. As each input has three fuzzy sets or membership functions in total, the total number of rules is 3*3*3=27. The first rule from Table 6-3 says " if the (Driver Past Behaviour (DPB) is **Good)** AND (Severity of Incident (SI) is **Not Severe** (NS)) AND (RSU Confidence (RC) is **Low**)**,** then the Reward is **Low**.

Table 6-3: Fuzzy Rules Used for Reward

Rule	DPB	SI	RCS	Reward
1	Good	Not Severe	Low	Low
2	Good	Not Severe	Medium	Medium
3	Good	Not Severe	High	High
4	Good	Low Severe	Low	Medium
5	Good	Low Severe	Medium	High
6	Good	Low Severe	High	Very High
7	Good	High Severe	Low	High
8	Good	High Severe	Medium	Very High
9	Good	High Severe	High	Very High
10	Neutral	Not Severe	Low	Low
11	Neutral	Not Severe	Medium	Low
12	Neutral	Not Severe	High	Medium
13	Neutral	Low Severe	Low	Low
14	Neutral	Low Severe	Medium	Medium
15	Neutral	Low Severe	High	High
16	Neutral	High Severe	Low	Medium
17	Neutral	High Severe	Medium	High
18	Neutral	High Severe	High	Very High
19	Bad	Not Severe	Low	Very Low
20	Bad	Not Severe	Medium	Very Low
21	Bad	Not Severe	High	Low
22	Bad	Low Severe	Low	Very Low

23	Bad	Low Severe	Medium	Low
24	Bad	Low Severe	High	Medium
25	Bad	High Severe	Low	Low
26	Bad	High Severe	Medium	Medium
27	Bad	High Severe	High	High

Table 6-4: Fuzzy Rules Used for Punishment

Rules	DPB	SI	RCS	Punishment
1	Good	Not Severe	Low	Very Low
2	Good	Not Severe	Medium	Very Low
3	Good	Not Severe	High	Low
4	Good	Low Severe	Low	Low
5	Good	Low Severe	Medium	Low
6	Good	Low Severe	High	Medium
7	Good	High Severe	Low	Medium
8	Good	High Severe	Medium	High
9	Good	High Severe	High	High
10	Neutral	Not Severe	Low	Low
11	Neutral	Not Severe	Medium	Low
12	Neutral	Not Severe	High	Low
13	Neutral	Low Severe	Low	Low
14	Neutral	Low Severe	Medium	Medium
15	Neutral	Low Severe	High	Medium
16	Neutral	High Severe	Low	Medium
17	Neutral	High Severe	Medium	High
18	Neutral	High Severe	High	Very High
19	Bad	Not Severe	Low	Very Low
20	Bad	Not Severe	Medium	Low
21	Bad	Not Severe	High	Low
22	Bad	Low Severe	Low	Low
23	Bad	Low Severe	Medium	Medium
24	Bad	Low Severe	High	High
25	Bad	High Severe	Low	Very High
26	Bad	High Severe	Medium	Very High
27	Bad	High Severe	High	Very High

6.4.3 Fuzzy Inference

Human level decision-making can be approximated with fuzzy inference. Fuzzy inference produces the output fuzzy sets from the input fuzzy sets. During fuzzy inference, each rule executes sequentially to obtain the desired output fuzzy set. A rule executes when its antecedent is satisfied. The antecedent of each rule is formed using fuzzy AND (min), fuzzy OR (max) and fuzzy NOT (inverse). In this application only the fuzzy AND and fuzzy OR are used as fuzzy logical operators. Fuzzy AND returns the minimum of all membership values from the antecedent part whereas the Fuzzy OR returns the maximum to clip or bound the height of output membership function. For example, let, the degree of belonging or membership value be denoted as μ and $\mu_{(DPB=neutral)}=0.26$, $\mu_{(SI=not\ severe)}=0.35$, $\mu_{(RCS=low)}=0.33$,

$\mu_{(DPB=bad)}$=0.74, $\mu_{(SI=less\ severe)}$=0.65, $\mu_{(RCS=medium)}$=0.67, $\mu_{(DPB\ =good)}$=0, $\mu_{(SI=high\ severe)}$=0, $\mu_{(RCS=high)}$=0. If we apply the fuzzy AND operator, then $\mu_{(DPB \cap SI \cap RCS)} = \boldsymbol{min}\ [\mu_{(DPB)}, \mu_{(SI)}, \mu_{(RCS)}] = \boldsymbol{min}$ [0.52, 0.52, 0.33]= 0.33. If the fuzzy logical operator is OR, then $\mu_{(DPB\ U\ SI\ U\ RC)} = \boldsymbol{max}\ [\mu_{(DPB)}, \mu_{(SI)}, \mu_{(RCS)}] = \boldsymbol{max}$ [0.52, 0.52, 0.33]= 0.52.

The returned value from the max or min, i.e., 0.33 or 0.52 is used to clip the height of the output membership function. They regulate the corresponding degree of membership value of the consequent part of each rule. This helps us to reshape or clip the output membership function based on the firing strength (output of the max/min) of the antecedent. When multiple output membership functions are available, then aggregation is used to obtain a combined output membership function from the individual output membership functions. This is done by combining all similar output fuzzy sets from the consequent part of all rules. In this way, the aggregated fuzzy output membership function is obtained. This is still a combined fuzzy set which is not a number. Hence, a defuzzification method is used to obtain a crisp value from this combined characteristic. Centre of gravity (COG) is the most widely accepted defuzzification method to find the final defuzzified value which is used by the RSU application to reach a final output value.

6.4.4 Defuzzification

This is the last step of the fuzzy control process. This step takes the aggregated output fuzzy membership set and produces a single crisp number which is our desired output from the fuzzy system. The centroid is the most widely defuzzification method of Mamdani inference. It delivers a point where a vertical line divides the aggregated output fuzzy set into two equal masses as expressed as Eqn (6-3).

$$CoG = \frac{\int_b^a \mu_A(x) x dx}{\int_b^a \mu_A(x) dx} \tag{6-3}$$

This method finds a point representing the centre of gravity of the fuzzy set, A, on the interval [a, b]. A reasonable estimate can be obtained by sampling a set of points.

6.4.5 Reward / Punishment Mechanism

The punishment or the reward level depends on the input combination which is possibly different in most of the disputes. As the DPB and RCS are possibly different for the same conflict. So the reward or punishment level should vary for the same dispute. If the punishment level is higher than the reward level for the same scenario, then the driver should be cautious about their future behaviour in the VANET which may deter them from launching more attacks. This means, to maintain good trust score, they need to behave honestly. Otherwise, their trust score will be dropped. This discourages malicious driver from launching future attacks due to the fear of loss of trust and being isolated. The system should

reach a decision based on how bad or good a driver was in the recent past or how much support / contradict an announcement receives in the form of feedback from clarifiers or how severe the incident is. Technically, the AND operation produces the lowest height among the antecedent values and the minimum of them is used as the height of the consequent. Thus, the AND operator selects less area from the output membership function when we combine them from all the executed rules. The subsequent centroid method yields a lower defuzzified value which is used for rewards. In contrast, OR selects a higher area compared to the AND operator when the output fuzzy sets are combined. Thus, this gives a higher defuzzified value when centroid defuzzification is applied. The punishment variable obtains this fuzzy value that a driver receives upon losing a dispute. This is typically seen from the simulation results where the punishment level is higher in most cases, but the reward level is better in only a few cases when different combinations of three parameters are considered. There are five fuzzy sets (linguistic variables) for the output membership function which are "Very Low", "Low", "Medium", "High", and " Very High" as shown in Fig. 6-5. For example, for punishment = 0.03, a vertical line touches only **Low** fuzzy set at 1, so $\mu_{(\text{punishment=Low})}$=1. Similarly, when reward = 0.08, $\mu_{(\text{reward =high})}$=0.5 and $\mu_{(\text{reward = very high})}$=0.5 since the vertical line for reward intersects with two fuzzy sets which are **High** and **Very High** at 0.5.

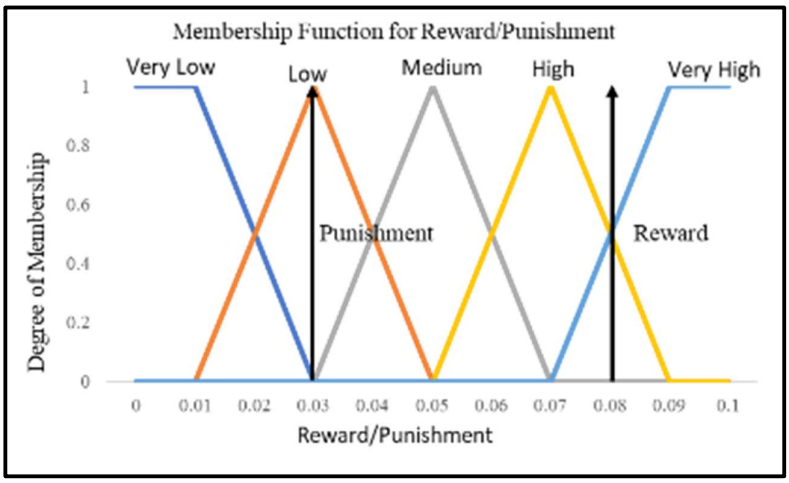


Fig. 6-5: Output Membership Function for Reward and Punishment

6.4.6 RSU Fuzzy Reward Assessment

In this section, we describe the fuzzy process to determine a single fuzzy output which we call the fuzzy reward. Fuzzy membership functions and rules for fuzzy reward are first used to obtain the output fuzzy sets. These fuzzy sets are combined in the aggregation step. Finally, the centroid defuzzification technique is used to obtain the desired output from the system.

6.4.6.1 Fuzzy Inference for Reward Assessment

In Fig. 6-6, the execution of some rules is shown based on the input combination during fuzzy inference. The antecedent part of the rules is evaluated first to generate an output from each rule with the height defined by the min or Fuzzy AND operation of the antecedent. The following inference considers DPB=0.8, SI=4, and RCS=0.33.

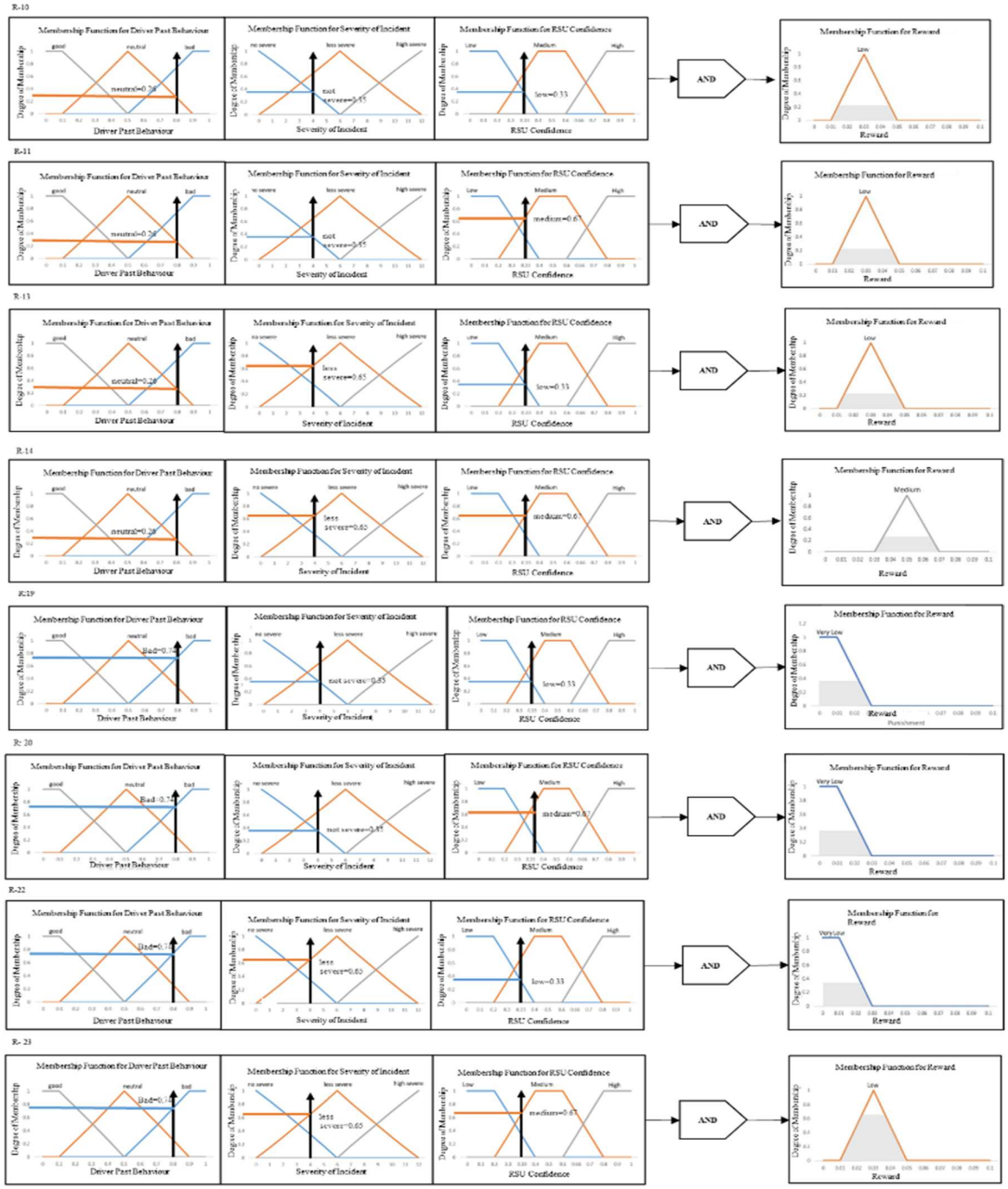


Fig. 6-6: Fuzzy Rule Inference for Reward Assessment

6.4.6.2 Redundant Rule Reduction for Reward

When multiple rules produce the same output fuzzy set with different values, then they are combined by taking the maximum of all consequent values for the same output fuzzy set; As Rules 10, 11, 13, and 23 have **Low** output fuzzy set, taking the maximum gives us Rule 23 with 0.65 as the membership degree for the **Low** output fuzzy set. As there is only one **Medium** fuzzy set, it is included directly. Also, Rules 19, 20, and 22 are associated with the **Very Low** output fuzzy set, thus the maximum consequent value from these three rules gives us Rule 20 to be included in the selected group for aggregation. This process is depicted in Fig. 6-7.

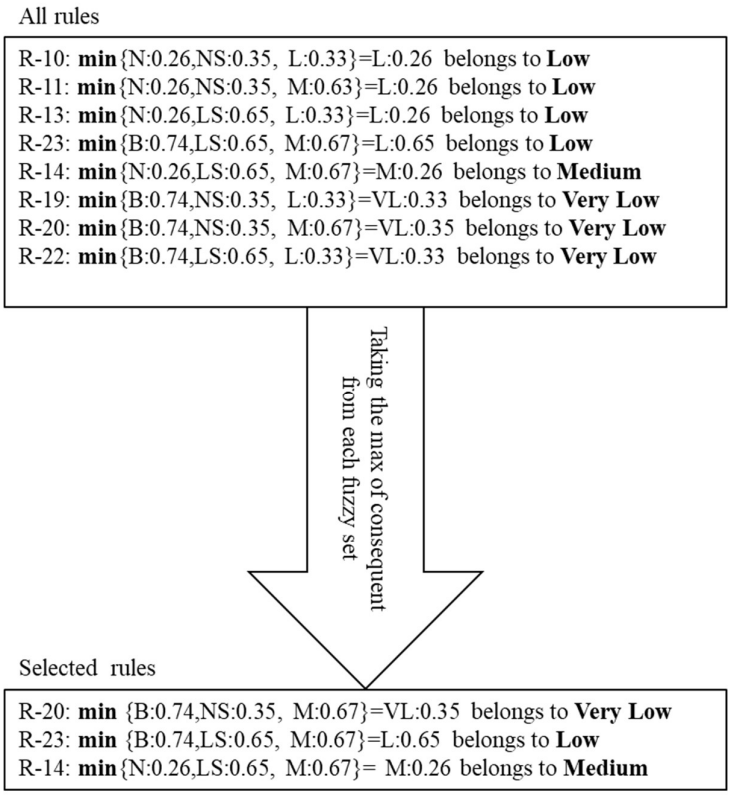


Fig. 6-7: Redundant Rule Reduction for Reward

6.4.6.3 Aggregation of the Consequents for Reward

Aggregation is applied to the selected rules which merges them to obtain a combined output membership function. In this step, only the output fuzzy set with the highest degree of membership is used when all the output fuzzy sets with lower values are inclusively covered. This is depicted in Fig.6-8.

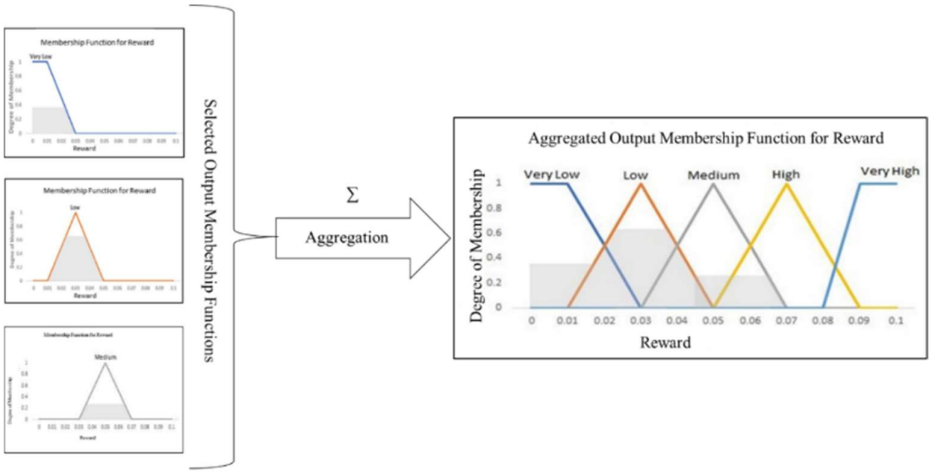


Fig. 6-8: Aggregated Output Membership for Reward Assessment

6.4.6.4 Defuzzification for Reward

Fig. 6-9 shows the application of centroid defuzzification. First, the area is sliced equally as shown. Then the centroid technique is applied to calculate the reward point at 0.030014, shown with a green arrow on the x-axis.

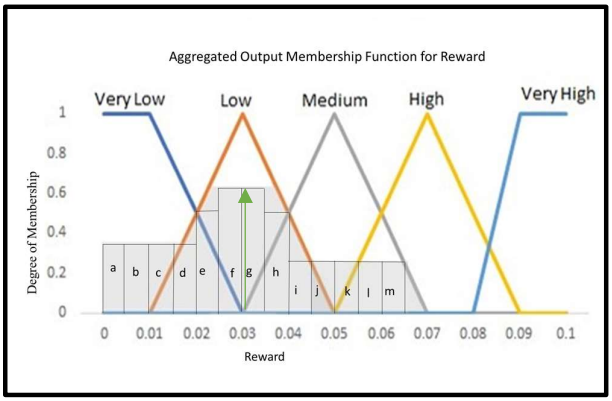


Fig. 6-9: Defuzzified Reward

The fuzzy reward R is calculated using the centroid technique as follows: first, we add all the x-coordinate values of the grey sliced area which have similar membership degree and then this sum is multiplied by their y-coordinate height. In this way, A1 is computed from the sliced area a, b, c, and d. Similarly, A2 is calculated from the area marked as e only, A3 is from f and g, A4 is from h, and A5 is from i, j, k, l, and m. Therefore,

A1 = (0+0.005+0.01+0.015+0.02) * 0.35

A2 = (0.02+0.025) * 0.48

A3 = (0.025+0.03+.035) * 0.65

A4 = (0.035 + 0.04) * 0.48

A5 = (.04+.045+.05+0.055+0.06+0.065) * 0.26

After this, we will calculate the denominator. To do this, we count all the equivalent y-values (height of the sliced area) from left to right as we slice the grey area and then we stop when we encounter a different y-value. This process repeats until the last sliced area is reached. For each equivalent y-coordinate obtained from the previous step, the number of sliced areas is counted in sequence. Then this sum is incremented by 1 to be multiplied by the y-value (height of the sliced area) to get an element to use in the denominator. In this way, we get, B1=0.35*5, B2=0.48*2, B3=0.65*3, B4=0.48*2, B5= 0.26*6. Finally, we obtain the fuzzy reward of 0.030014 as follows:

R = (A1+A2+A3+A4+A5)/(B1+B2+B3+B4+B5)

R = 0.2155/7.18 = 0.030014

6.4.7 RSU Fuzzy Punishment Assessment

In this subsection, the fuzzy process to determine one fuzzy output we call the fuzzy punishment. Fuzzy membership functions and rules are used to obtain the output fuzzy sets. These fuzzy sets are combined in the aggregation step. Finally, centroid defuzzification is used to determine the appropriate punishment from this system.

6.4.7.1 Fuzzy Inference for Punishment Assessment

This is an example fuzzy inference for punishment assessment with DPB = 0.8, SI = 4 and RCS = 0.33. For clarity, only a subset of the rules are included in this fuzzy inference though there were 26 rules fired based on the input combinations. Thus, Fig. 6-10 shows just some of the rules executed for punishment execution at an RSU using the values from Table 6-1.

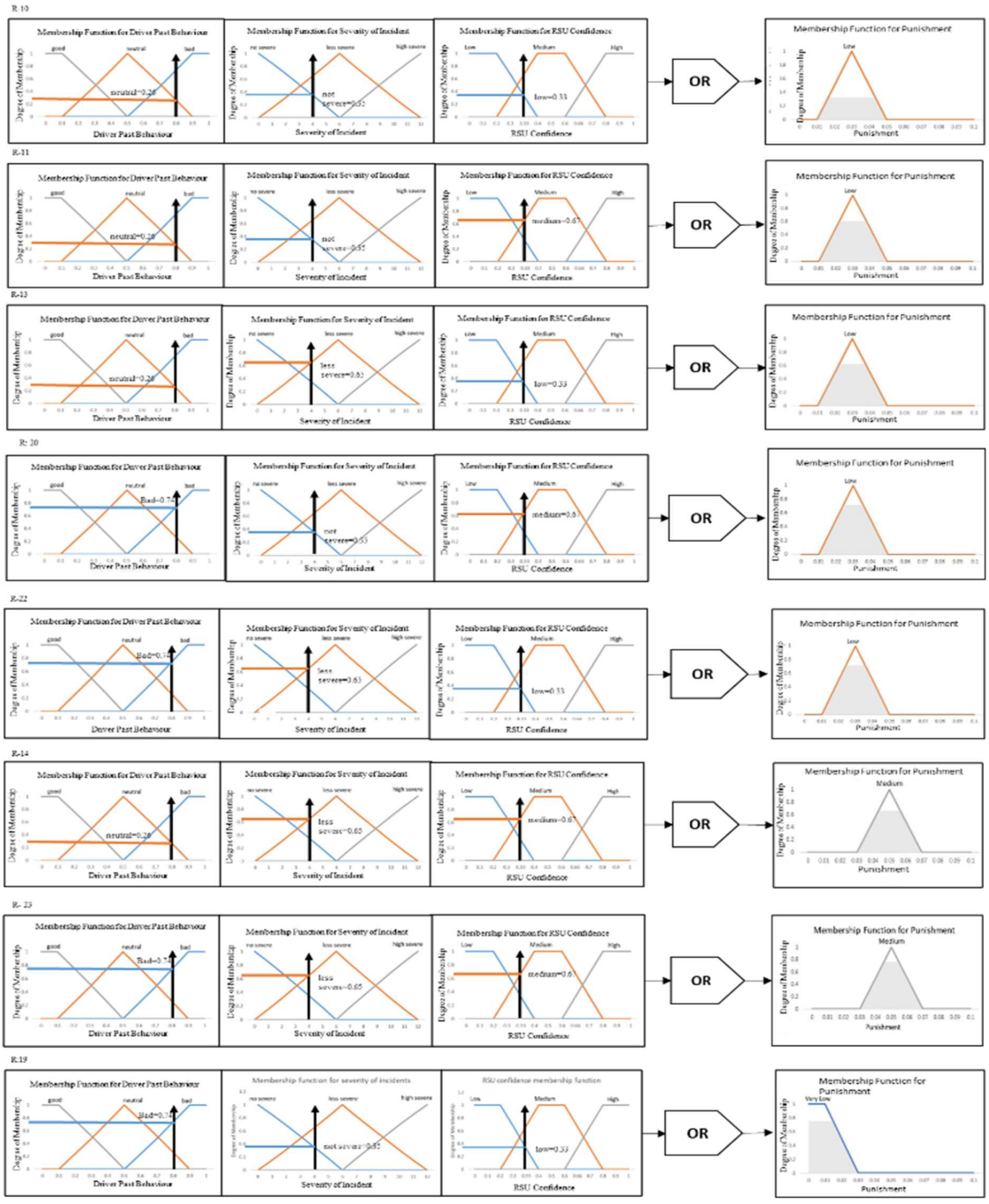


Fig. 6-10: Fuzzy Rule Execution for Punishment Assessment

6.4.7.2 Redundant Rule Reduction for Punishment Assessment

In this step, the rule with the maximum consequent value is selected for the aggregation step from the set of rules which are associated with the same output fuzzy set. This process is shown in Fig. 6-11. There are five rules associated with the **Low output** fuzzy set. Selecting the maximum consequent value from the **Low** fuzzy sets yields Rules 20 and 22 to be included. Since, they both have the same value, either of them can be selected. Thus, we include Rule 20 randomly in the selected rule group.

The maximum consequent value from the two **Medium** output fuzzy sets selects Rule 23 to be included. Lastly, there is only one fired rule associated with the **Very Low** output fuzzy set, which is added directly.

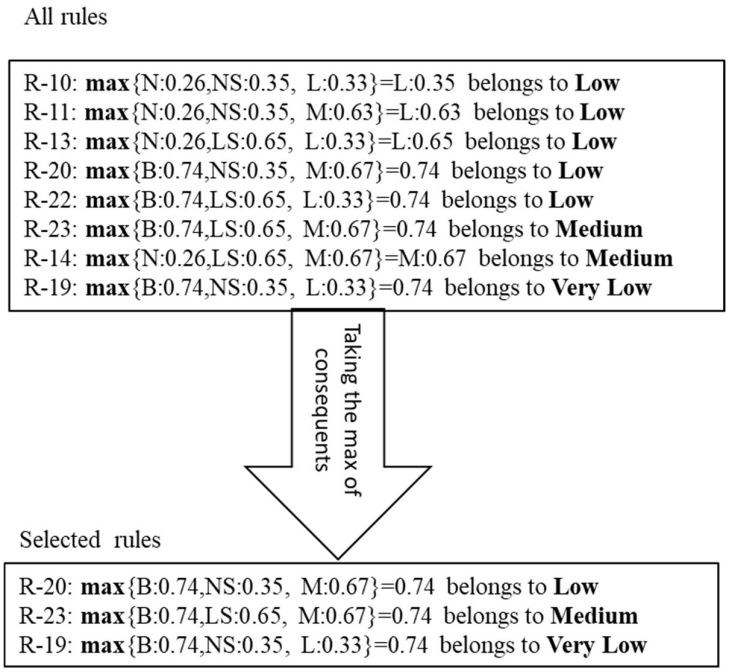


Fig. 6-11: Redundant Rule Reduction for Punishment Assessment

6.4.7.3 Aggregation of the Output Fuzzy Sets for Punishment Assessment

Aggregation is the process of combining clipped output membership functions into a single output membership function for each output variable. The aggregated output fuzzy membership function is shown in Fig. 6-12 for punishment assessment. The selected area is shaded grey in the aggregated output membership function.

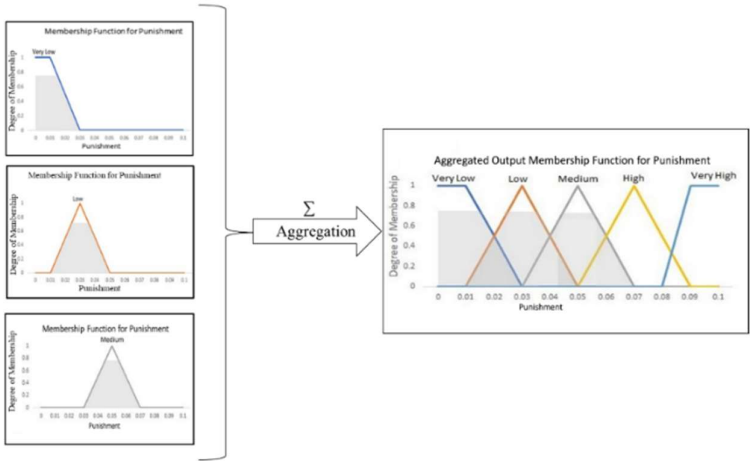


Fig. 6-12: Output Membership Aggregation for Punishment

6.4.7.4 Defuzzification for Punishment

Fig. 6-13 shows the application of centroid defuzzification. First, the area is sliced equally as shown. Then the centroid technique is applied to determine the punishment point at 0.032067, shown with a red arrow on the x-axis.

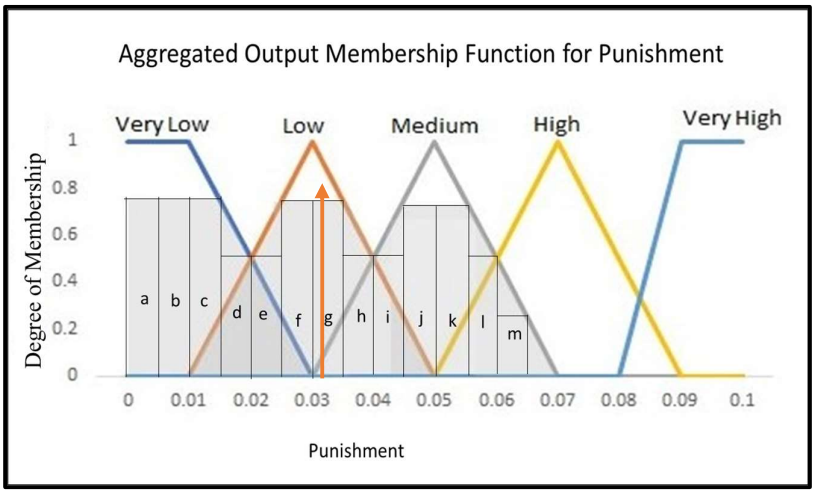


Fig. 6-13: Defuzzified Punishment

The fuzzy Punishment P is calculated using the centroid technique as follows: first, we add all the x-coordinate values of the grey sliced area which have similar membership degree and then this sum is multiplied by their y-coordinate height. In this way, A1 is computed from the sliced area a, b, and c. Similarly, A2 is calculated from the area marked as d and e, A3 is from f and g, A4 is from h and i, A5 is from j and k, A6 is from l and A7 is from m. Therefore,

A1 = (0+0.005+0.01+0.015) * 0.74

A2 = (0.015+0.02+0.025) * 0.5

A3 = (0.025+0.03+0.035) * 0.74

A4 = (0.035+0.04+0.045) * 0.5

A5 = (0.045+.05+0.055) * 0.74

A6 = (0.055+0.06) * 0.5

A7 = (0.06+0.065) * 0.3

We then count all the equivalent y-values (height of the sliced area) from left to right as we slice the grey area and then we stop when we encounter a different y-value. This process repeats until the last sliced area is reached. For each equivalent y-coordinate obtained from the previous step, the number of sliced areas is counted in sequence. Then this sum is incremented by 1 to be multiplied by the y-value (height of the sliced area) to get an element to use in the denominator. In this way we get, B1=4*0.74, B2=3*0.5, B3=3*0.74, B4=3*0.5, B5=3*0.74, B6=2*0.5, and B7=2*0.3. Finally, we obtain the fuzzy punishment of 0.032067 as follows:

P = (A1+A2+A3+A4+A5+A6+A7)/(B1+B2+B3+B4+B5+B6+B7)

P = 0.3848/12 = 0.032067

6.5 Implementation

The fuzzy RSU scheme is implemented in MATLAB 2022. There is a built-in fuzzy logic designer app in which three inputs are created along with their input membership functions, and corresponding fuzzy sets. Two different sets of rules are entered into the rules editor and all the rules are given equal weight. As this is a two-output fuzzy system, two output membership functions and corresponding fuzzy sets are also created. The fuzzy OR operator is used for the punishment and the fuzzy AND is applied to reward assessment. There are three fuzzy sets for each input and five fuzzy sets for each output. During fuzzy inference arising from a dispute, all twenty-seven rules are evaluated individually to produce fuzzy output sets for each output. Aggregation is applied on these output fuzzy sets and then the centroid method is used on the combined fuzzy sets to return the desired fuzzy reward and punishment. These output values from MATLAB are processed and inserted into two different sets of lists in OMNeT++ to be used with the proposed model. There are eleven possible values of DPB. Hence, for each DPB value, all possible values of SI and RSU confidence are considered. In this way, all different combinations of input values are used with the fuzzy system. For each DPB value, a different data structure is created in OMNeT++ to improve the search speed.

When a dispute decision is ready, an RSU asks for the DPB data from the TA. The RSU calculates the DPB for the drivers concerned. The RSU also calculates the confidence score of the drivers from the collected feedback and additionally determines the severity level of event. Then, the RSU obtains the corresponding fuzzy reward and fuzzy punishment from the lists. These values are directly used in the reward and punishment messages which the RSU announces and forwards to nearby RSUs to be announced by them. In this way, the respective driver/vehicle receives the fuzzy RSU reward or punishment.

The following set of experiments uses a Markov chain-based driver behaviour model which is implemented inside the TPD of every regular vehicle. This model governs a driver's announcement behaviour by setting the probability of sending trustworthy or untrue messages from different trust states.

6.6 Fuzzy versus Fixed RSU Judgement

6.6.1 Simulation Setup

The fuzzy reward / punishment method is applied when dispute decisions are ready at RSUs. A comparison is made between the fuzzy versus fixed reward and punishment. To this end, a series of experiments is conducted to evaluate their performance. The trust framework, the fuzzy reward and punishment scheme, and a Markov state transition driver model are implemented in Veins which

comprises OMNeT++ and SUMO. The participating vehicles run for 5000 simulation seconds on a fixed circular route in the Erlangen city map (Sommer et al [17]). 100 vehicles are added at the beginning of the simulation and their numbers remain constant. Vehicles undergo a warm-up period where all vehicles run without announcing any events. When the warm-up period has elapsed, a fixed sender driver announces messages periodically at 1000s intervals starting from 500s. Multiple types of event announcements are considered from the same driver for behaviour analysis. The events are scheduled as an accident message at 500s, debris message at 700s, road defect message at 900s, traffic element problem message at 1100s and tree on road message at 1300s. When they are announced, a fixed set of reporter drivers of vehicles V[1], V[2], V[3], V[4], and V[5] deterministically send untrue attack reports after their reception. The trust data is recorded separately for the fuzzy and fixed systems. The trust framework has other mechanisms rewards and punishments (i.e. via the TPD) which are omitted for clarity since fuzzy logic is used to improve the RSU reward and punishment assessment only. Updates of trust from the RSU fuzzy reward and punishment unit are recorded on graphs to compare with trust updates from the fixed scheme. Two density distributions from the trust data (one uses the initial trust data, and another uses the trust data when the simulation ends) are plotted for both experiments. Table 6-5 shows the experimental parameters. Some of the parameters are defined in the trust framework and some are selected from observations. For example, the announcement reward, clarifying, and relaying rewards are defined in the trust framework. As all vehicles take 500 simulation seconds to join the simulation, thus the warm-up period is 500s. Fuzzy assessment of the driver behaviour generates varied amounts of reward/punishment to the drivers, thus it is labelled "varied" in the table. This set of experiments runs for 5000s. because we obtain sufficient data within this period. As there are five different events announced starting from 500s and each event announcement is separated by 200s. Different events are announced at 700s, 900s, 1100s, and so on. Consequently, the same event is announced periodically with 1000s interval. We have used RSUs and a TA for these experiments. Table 6-6 shows the trustworthy and untrue announcement probabilities for the different trust states of the driver model which defines the behaviour of sender driver.

Table 6-5: Simulation Parameters

Parameters	Values
Fuzzy reward and punishment	Varied
Fixed reward and punishment	0.1
Data collection nature	1. When all features enabled 2. When only RSU judgement applied
Simulation period	5000s
Warm-up period	500s
Periodic event announcement	1000s
Initial trust	Uniform distribution (0.5-0.6)
Number of vehicles	100
Multiple types of events generated	Five different events
Number of RSUs	12
Number of TA	1
Attacker model	Untrue and inconsistent behaviour

Untrue attack generation	Based on the message class
Announcement reward	Maximum of 0.08 (0.01 to 0.08 based on delay and distance)
Clarifier reward	0.08
Relaying reward	0.002

Table 6-6: Message Announcement Probabilities

Trust State	Probability of Malicious Message	Probability of Trustworthy Message
"Very Good"	20	80
"Good"	40	60
"Normal"	50	50
"Bad"	80	20
"Very Bad"	100	0

6.6.2 Performance Comparison of RSU Fuzzy and Fixed Reward / Punishment

6.6.2.1 Scenario 1 – Trust Update from Fuzzy RSU Reward / Punishment

Fig. 6-14 shows the trust score evolution for six vehicles only. This experiment only shows the trust evolution from the RSU judgements to demonstrate drivers lying or honest behaviour. When there is a dispute, a driver earns a reward, and another driver gets punishment. Hence, trust evolution of the other 94 vehicles is not shown in Fig. 6-14 though there are 100 vehicles participating in the experiment. The x-axis represents simulation seconds, and the y-axis shows the updated trust from the fuzzy RSU unit. During this experiment, the driver of V[0] sends scheduled events periodically. The initial trusts are assigned from a uniform distribution with the range of 0.5 to 0.6. The driver of V[0] starts with a "normal" trust state which governs his/her behaviour in message announcements. This state is configured to send malicious and trustworthy messages equally in the state transition model.

It is seen that V[0] builds trust from the fuzzy rewards as it announces only trustworthy messages while the reporters get fuzzy punishments which reduces their trust as the simulation progresses. First, V[0] moves to "Good" state and then to "Very Good" states. V[0] reaches the maximum trust at about 1800s with "Very Good" state. Alternatively, reporters in this experiment send untrue reports and move from the "normal" to the "bad" trust state. For example, the driver of V[2] always sends false reports and receives RSU punishments. His/her trust score plunges to the lowest value of 0.34 at 2900s due to being malicious. It is noticeable that the first reward of V[0] is highest as the driver has no punishment records in the DBP whereas the latter judgements are not seen as high as the first one. Since some latter rewards are from the disputes relating to the less severe announcements. Alternatively, the fuzzy RSU punishments are not very harsh initially which is seen in the reporter vehicles V[1] and V[5]'s punishments. They increase slightly in the later punishments where the severity of incident, punishment records in the DPB, and RSU confidence influence the outcome. In later disputes, event severity levels are different which vary the punishment. Hence, the trust increment varies throughout the experiment

whereas in the fixed reward scheme trust increments / decrements are fixed irrespective of mitigating factors. So, with the fuzzy scheme, a driver has more chances to improve trust scores from subsequent announcements and trustworthy reporting. This way their network participation lifespan is extended. Fig. 6-15 depicts the trust scores of all vehicles which participated in this experiment. It is noticeable from this Fig. that the trust scores of most vehicles are unchanged throughout the simulation as they do not report or announce any messages and there is no forwarding or clarifying reward for others.

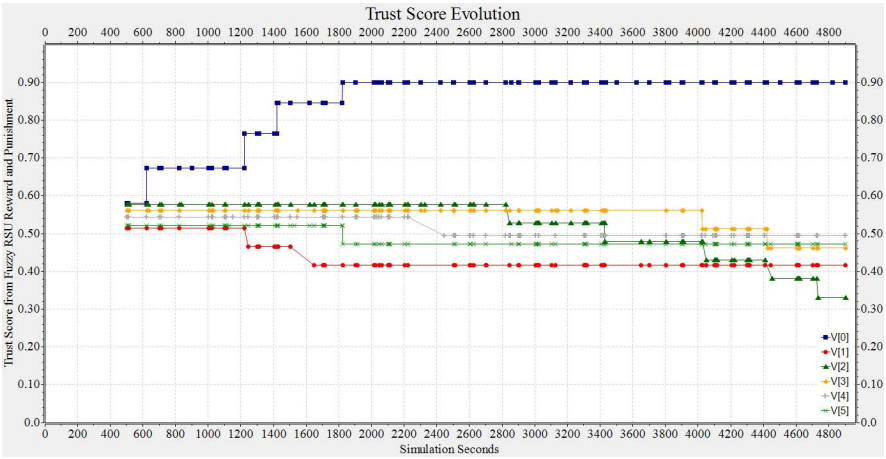


Fig. 6-14: Trust Score Evolution from the Fuzzy Reward and Punishment

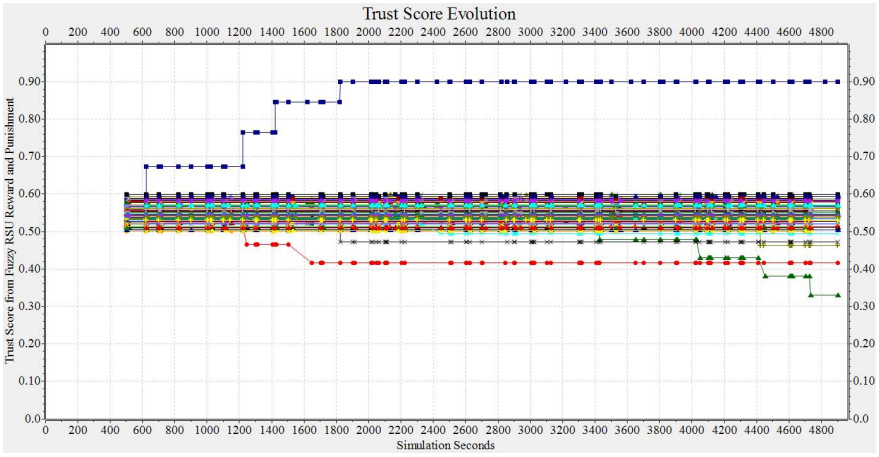


Fig. 6-15: Trust Score Evolution of All Vehicles Using Fuzzy Reward and Punishment

The two curves in Fig. 6-16 show the density distribution of trust scores of vehicles which are collected at the beginning and the end of the simulation. The initial trust score of all vehicles is between 0.5 to 0.6. The right-hand chart in Fig. 6-16 shows that V[0] reaches 0.9 which is marked by a dot. Most of the vehicles do not see any trust score alterations apart from the three vehicles (trust is about 0.4) which are the reporters in this experiment. This is because they do not engage in any disputes from

which they can earn or lose trust. Additionally, they are not given any reward from forwarding or other activities. The long gap in the right-hand chart means no vehicle other than V[0] achieves this score due to the experimental design and this result is expected. Also, the driver behaviour model governs their honest and dishonest announcements.

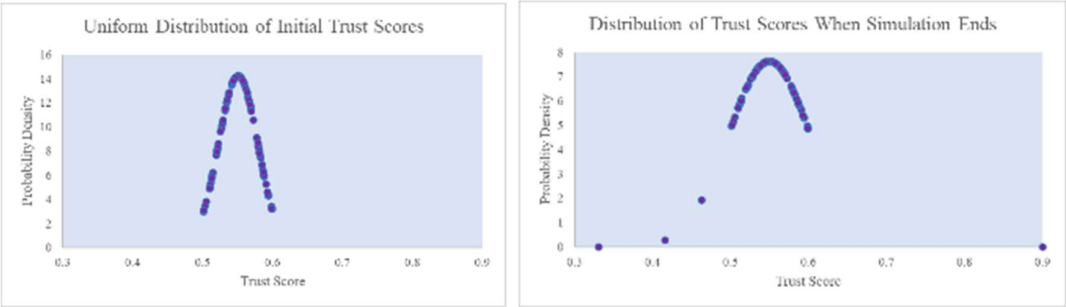


Fig. 6-16: Distribution of Trust Scores at the Beginning and End of the Experiment

6.6.2.2 Scenario 2 – Trust Update from Fixed RSU Reward / Punishment

This experiment measures the trust score of vehicles from the fixed RSU reward and punishment scheme. In Fig. 6-17, simulation time is on the x-axis and the y-axis shows the trust score. This is conducted with the set of parameters defined in Table 6-5, but the RSU judgement is fixed (0.1) for every driver. V[0] sends a malicious message initially and gets RSU punishment that reduces its trust to less than 0.5. From this stage, it is configured to send more malicious messages 80% of time. Thus, its trust subsequently decreases due to RSU punishments. When its trust score belongs to "very bad" state, it sends only malicious messages. In this way, its trust is reduced to 0.05 which meets the condition to block its access. Alternatively, the reports from V[1] win all disputes and hence its characteristic always shows an upward trend. Also, V[4] and V[5] win two other disputes over V[0] and hence receive RSU rewards which improve their trust. It should be noted that there is no activity after 4400 seconds as all events are completed by this time. Fig. 6-18 shows the trust scores of all participants in this experiment.

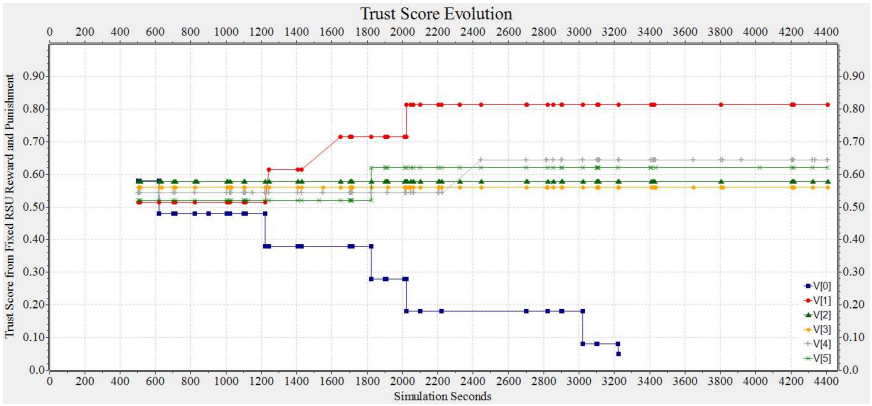


Fig. 6-17: Trust Score Evolution from the Fixed RSU Reward and Punishment

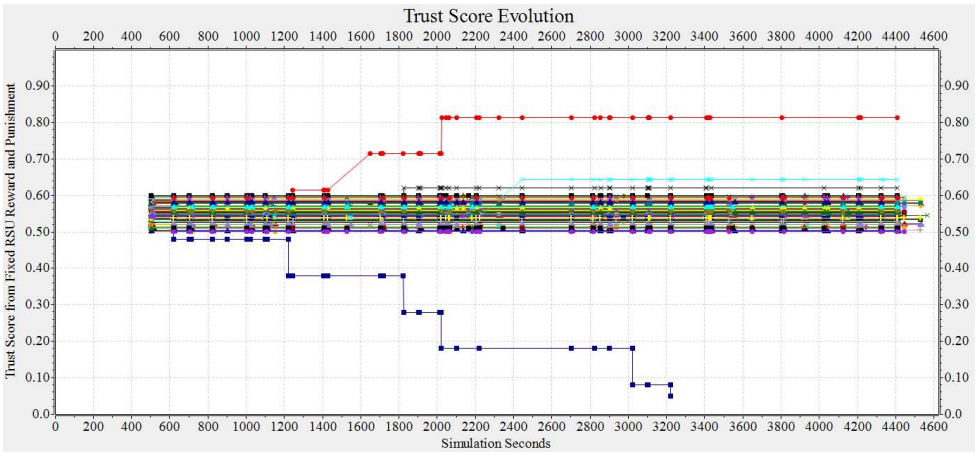


Fig. 6-18: Trust Score Evolution of All Vehicles with RSU Fixed Reward / Punishment

Fig. 6-19 shows the initial trust and the final trust distribution in two density curves. The first density chart shows the trust scores of all vehicles generated from a uniform distribution. However, the right-hand chart plots the trust scores of all vehicles when the simulation ends. As expected, in the second chart, the trust of most vehicles is unchanged as they do not engage in any disputes from which their trust can change. The right-hand chart confirms some vehicles with positive behaviour build their trust from truthfully reporting activities whereas sender V0 is access-blocked, leaving its trust at 0.05. With this fixed RSU judgement, vehicles have less opportunity to modify their behaviour and vehicle access-blocking is more likely as shown in the right chart in Fig. 6-19.

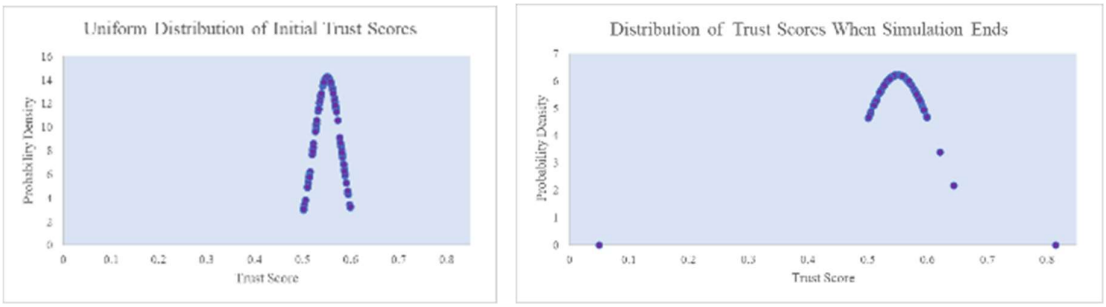


Fig. 6-19: Distribution of Trust Scores at the Beginning and End of the Experiment

6.6.3 Performance Comparison of Trust Scores When All Rewards and Punishments are Allocated

6.6.3.1 Scenario 1 – Trust Updates using Fuzzy RSU Reward / Punishment

This experiment is conducted by enabling all rewards and punishments with the original framework. Fig. 6-20 plots the reward and punishment from the announcements, relaying reward, and the fuzzy RSU reward and punishment. The chart shows only the trust score evolution of six vehicles. The x-axis

shows elapsed simulation time, and the y-axis shows the trust score of these vehicles. Overall, all vehicles develop trust more rapidly than the previous experiment as vehicles receive rewards from all kinds of activities. There is a fluctuation of V[0]'s trust for punishment at 1200s and after that, the trust characteristic remains steady throughout the remaining simulation. Reporters show both positive and negative behaviours as they receive announcement and relaying rewards besides punishments. As all other rewards are enabled, there are some sudden changes of trust noticed. Although, these decremental steps do not last long; rather they return to the peak trust score when vehicles announce or become a clarifier or reporter. Fig. 6-21 includes the trust records of all vehicles. It is clearly visible that all vehicles improve trust in general than the earlier fuzzy experiment even though they are neither senders nor reporters. However, the fuzzy rewards and punishments are only given to the vehicles V[0], V[1], V[2], V[3], V[4], and V[5].

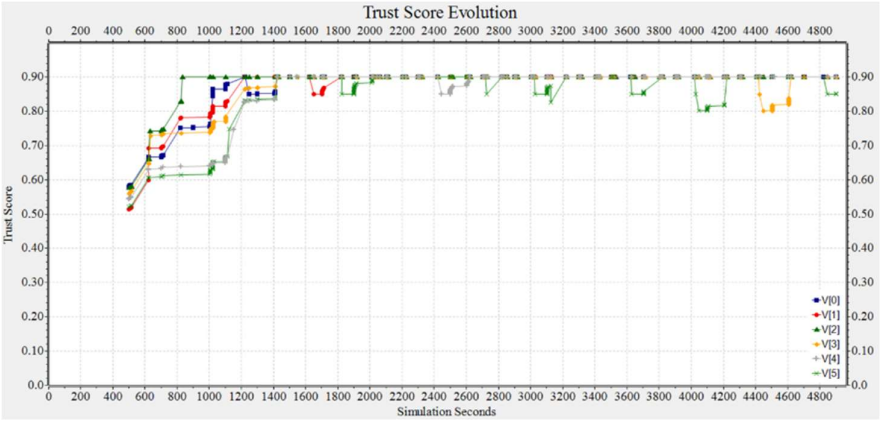


Fig. 6-20: Trust Score Evolution of Some Vehicles Using All Rewards / Punishments

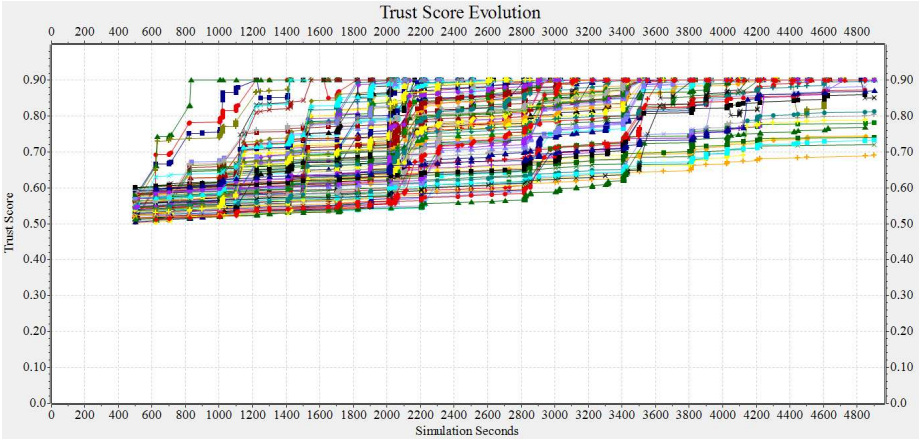


Fig. 6-21: Trust Score Evolution of ALL Vehicles Using All Rewards / Punishments

Fig. 6-22 depicts the density distribution of trust scores of all vehicles at the start and end of the simulation run. This experiment allocates all forms of reward and punishment besides the fuzzy assessment as vehicles can earn rewards from announcing, reporting, clarifying, and forwarding. Thus, all of them improve their final trust over the simulation period which results in their trust being at least 0.68. Among them, many vehicles also reach the maximum trust of 0.9.

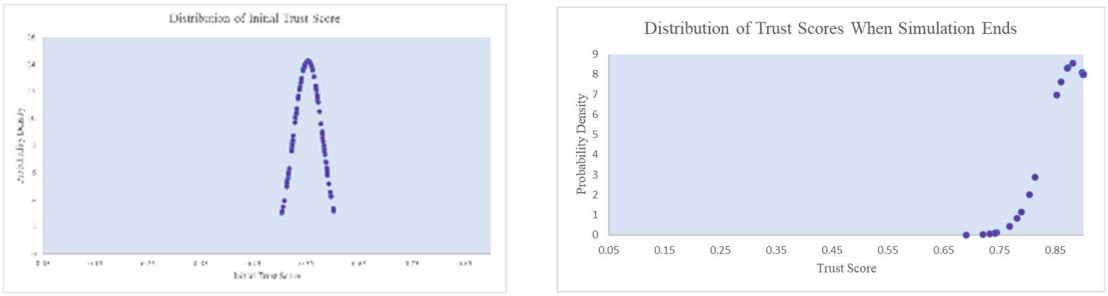


Fig. 6-22: Distribution of Trust Score With All Rewards and Punishments

6.6.3.2 Scenario 2 – Trust Updates from the Fixed RSU Reward / Punishment

Fig. 6-23 shows the trust for vehicles which actively communicate during the simulation. The rewards and punishments are fixed (0.1) for this experiment and vehicles receive all forms of reward and penalty. Many abrupt ups and downs are noticed than the previous experiment because the magnitude of RSU judgements are higher. V[0] is trustworthy throughout the simulation and its trust score reaches 0.9 at 1000s. V[1] and V[5] show the most dramatic changes in their trust as they send reports mostly. V[3] also observes some changes of trust throughout the simulation. In this chart, the decrement in trust relates to the RSU punishments only and V[0] never receives any RSU punishment. Finally, all these vehicles finish with 0.9 final trust scores when the simulation ends. Fig. 6-24 shows the trust of all participating vehicles when all the rewards and punishments are enabled. Overall, the trust builds faster compared to Fig. 6-21 as the reward magnitude is typically higher than in the fuzzy case.

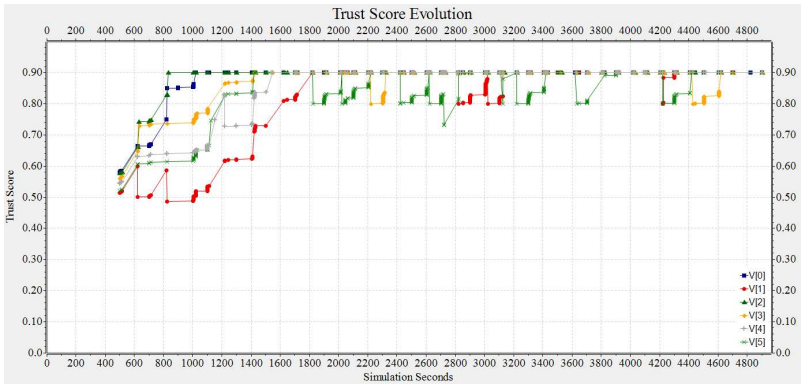


Fig. 6-23: Trust Score Evolution of Some Vehicles Using All Rewards / Punishments

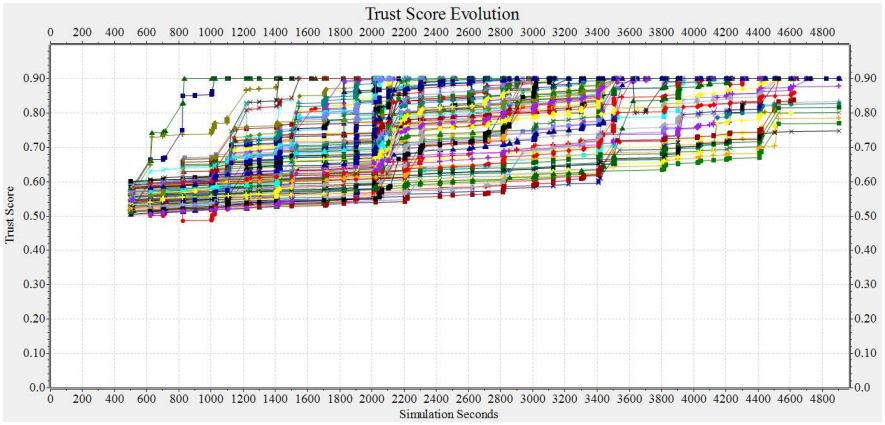


Fig. 6-24: Trust Score Evolution of ALL Vehicles Using All Rewards / Punishments

Fig. 6-25 depicts the trust score distribution of all vehicles at the start and at the end of this experiment. The initial trust of vehicles is between 0.5 and 0.6. After that, vehicles announce, forward messages and even report messages to earn rewards. The final trust is at least 0.75 and their trust increment is faster than for the fuzzy-based experiment. However, the assessment of reward and punishment does not consider the severity of events, RSU confidence, nor driver past behaviour.

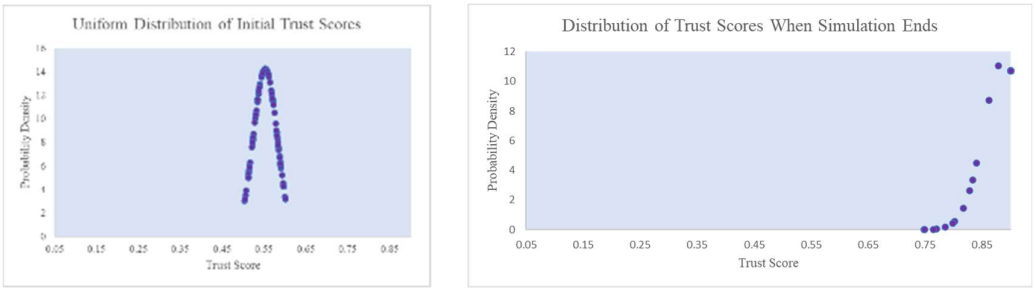


Fig. 6-25: Distribution of Trust Score With All Rewards / Punishments

6.6.4 Discussion on Results

It is seen that trust building is faster in the fixed RSU judgement system as it assigns a high amount (0.1) irrespective of event type and driver behaviour compared to the fuzzy system which provides an amount in the range 0.01 to 0.1 based on the fuzzy evaluation result. In the fixed reward and punishment system, when trust is developing, it reaches the peak trust or the lowest permitted value earlier. In contrast, with a fuzzy system, the allocation of reward and punishment amounts is more proportionate, so vehicles have more time to correct their future behaviour and continue normal operations. The blacklisting of a vehicle is also delayed when using the fuzzy system.

When only RSU rewards and punishments are given, in the fuzzy system, the sender vehicle reaches 0.9 trust, whereas the same sender vehicle is access-blocked with the fixed system. However, this is not only due to the magnitude of the judgement but also for the RSU decisions about the announcement to

be trustworthy or untrue. With the fuzzy system, a reporter vehicle reaches a low trust score though it has some trust left to carry more communication and it may correct its behaviour and achieve a good trust score later. Overall, with the fuzzy system justified levels of reward and punishment are given. When all forms of reward and punishment are given, more vehicles reach the highest trust with the fixed system as the size of the rewards and punishments are greater than for the fuzzy system. The lowest trust with the fixed system is 0.75 whereas it is 0.68 with the fuzzy system after running the experiment for 5000s under the same conditions. In the fuzzy system, the trust values are more stable than the fixed system in the sense that when trust is being built or lost it does not change dramatically. Additionally, the fuzzy system considers environmental dynamics for fuzzy judgements, e.g., event severity, driver past behaviour and confidence score which we believe is appropriate for reviewing the disputes.

6.7 Summary

In this chapter, we have presented the proposed Mamdani fuzzy logic-based RSU reward and punishment assessment scheme. This application considers event severity, driver past behaviour and RSU confidence to determine a justified level of reward or punishment for drivers. The reward and punishment mechanism uses a different set of rules to map the input to the output fuzzy set. A Markov-chain based driver behaviour model is used to control the announcement behaviour of drivers when conducting the series of experiments. Typically, good drivers are more likely to remain good and bad drivers, vice versa. Messages are announced in a more controlled manner than the set of experiments conducted in Chapter 4. They are reported only from a predetermined subset of vehicles after their reception which differs from the previous arrangement, where any vehicle can report on a sender announcement. First, we compare the fuzzy reward and punishment scheme to our fixed one, showing the evolution of trust scores and the initial and final trust density distribution curves. It is observed that the fuzzy system allows varying reward or punishment amounts to be allocated in each dispute based on the circumstances. Conversely, with the fixed system, in all disputes, the amount of reward or punishment is fixed which often results in drivers quickly experiencing access blocking or attaining a high trust score. We also considered a performance comparison between these two systems when all forms of reward and punishment are enabled. The results suggest that the fuzzy system achieves more stable trust dynamics throughout the simulation.

Chapter 7: Discussion, Conclusion, and Future Work

7.1 Discussion

In this research, we have developed a sender-side trust framework for VANETs which manages trust inside the Tamper-Proof Device (TPD) of each regular vehicle. This framework considers multiple types of vehicles, Road-Side Units (RSUs) and a Trust Authority (TA). Official vehicles are police, ambulance, and fire service vehicles. Regular vehicles are the primary user of this framework. Other types of vehicles can be added incrementally. This framework does not require any trust metric dissemination unless there is a dispute in which case the RSU collects feedback from both trusted clarifiers and official vehicles, if available. This reduces the communication overhead relative to existing receiver-side trust frameworks (Lu et al [6], Mühlbauer et al [14], and Chen et al [97]). In receiver-side schemes (Guleng et al [15], Wei et al [16], Huang et al [91], Wu et al [112], and Wei et al [113]), opinions and recommendations are collected from the neighbour vehicles and RSUs. In some reputation approaches (Mühlbauer et al [14], Li et al [93], and Pu et al [96]), a centralized server performs the trust computation to verify traffic events which increases both decision delay of receiver vehicles and network communication overhead. This delay can exacerbate undesirable situations. Conversely, our framework is not based on receiver evaluations and so receiver vehicles can typically trust the sender's announcements and make decisions without further communication. For any untrue message, the sender driver would be punished by an RSU when discovered by the reporting and dispute resolution mechanism. Due to the punitive measures associated with lying, we expect this process to be invoked rarely. In this way, the framework reduces the delay in decision-making (response time) and communication overhead compared to existing trust approaches.

Messages are grouped into tiers and different trust thresholds are attached to each tier. A driver can only announce an event from a tier when he/she has a sufficient trust score. In this way, trust score of the drivers regulates the announcement capabilities in the VANET. Drivers earn trust rewards from event announcements, clarification, untrue attack announcements, relaying, and beaconing. They lose trust from the execution of an RSU punishment and/or a TPD punishment. The TPD reward for an announcement is withheld for a period. When this period expires, if there is no complaint about the announcement, then the TPD adds the reward to the trust of the driver. The withhold period is long enough to ensure disputing drivers have time to respond. If there is no nearby vehicle, an RSU receives it and begins retransmissions. Later, when there is a vehicle which receives a message from the RSU that notices it is an untrue announcement then it can raise a complaint / report to the RSU. If it is received by the sender TPD within the withhold period, then it does not add the reward for the announcement. However, the TPD punishment from the announcement is executed promptly whenever a driver delays or travels more than a threshold value. Also, drivers receive a clarifying reward instantly from the TPD though the reward for untrue attack reporting is delayed until the RSU dispute ruling. The framework

blocks communication from malicious drivers when their trust score is too low. The access-blocking is equivalent to the blacklisting of other trust models. The RSU punishment is given incrementally for subsequent untrue announcements from the same driver even when the driver announces some untrue and true announcements. The trust framework blocks access of a driver when he/she announces untrue event three times.

When an RSU receives an untrue attack report, it informs nearby RSUs to prevent multiple initiations of the dispute resolution process. Each dispute is resolved by a single RSU only. While resolving a dispute, only trusted drivers can send a feedback message containing their opinion and trust score to determine which driver(s) have sent untrue messages. An RSU may receive multiple reports from different reporters. However, when an RSU sees that it is related to an ongoing dispute, it ignores repeated reports originating from different reporters because the repeated dispute is related to the same sender event. During this process, if an official vehicle sends feedback, then the RSU decides using this feedback alone. However, when a dispute cannot be settled due to insufficient feedback or when both sum of positive weighted replies and the sum of negative weighted replies are equal, an RSU saves it to a list. It then asks an official vehicle whenever it comes within the RSU coverage area to visit the event location and inform the RSU, if the severity of event is considered high. If the severity of event is not high, then it is not needed for an RSU to ask an official vehicle to visit and to inform. It may happen that the official vehicle does not see any symptom of the event or does not get any data from the surroundings, then it informs an RSU that the sender driver lied in the network. However, when the problem is resolved by the time the official vehicle arrives at the scene, it will try to collect data from the surroundings first (by authoritative inspection this may involve interrogating nearby people). When it is not possible, then it recommends the RSU not to reward or punish the drivers since it cannot prove the event.

Messages are organised into tiers based on their importance. Only highly trusted drivers can send announcements relating to events of significant magnitude. Drivers with a lower trust are unable to send such messages. This provides a measure of regulation in the framework, reducing the opportunity for malicious behaviour. Even so, a scheme is needed to detect and to punish malicious drivers to correct future behaviour. RSUs forward dispute results to the TA to save in the driver profile database. Additionally, RSUs forward any incident to the TA to save into the incident database.

The framework is tested using a set of possible road events as well as observing the message forwarding sequences to confirm that information is disseminated appropriately for each traffic event. The framework is verified to ensure it can detect untrue and inconsistent attacks and punish drivers as per the design. After this the accuracy of the framework is measured by changing the vehicle density and the clarifier feedback, using different ratios of malicious and benevolent opinions. The RSU uses feedback to classify untrue attacks as true positive, true negative, false positive and false negative cases

which are shown on 3D charts. The results show that the framework detects the cases in line with the feedback from the clarifiers. When trustworthy feedback outweighs malicious feedback, the classification yields the correct decision, as expected.

With this framework communication overhead and driver decision time can be reduced as sender-side evaluation is implemented. The framework is compared with a reputation-based receiver-side approach to verify this statement. A series of experiments are conducted with 10, 20, 30, 40, 50, 60, and 70 vehicles using the two alternate route scenarios. The outcome from the comparison indicates that the communication overhead is almost half when the vehicle density is low. With increasing vehicle density, the communication overhead remains at least two times lower than the baseline approach and in some scenarios this overhead is four times lower than the baseline. The response time of the receiver vehicles is effectively zero irrespective of the vehicular density with the proposed trust framework. However, the response time of the receiver vehicles in the baseline is set by the duration of a timer which determines a verification window.

The trust framework is investigated further to find weaknesses for potential improvements. It is found that drivers are blocked very early when they announce untrue events (i.e. three within a defined interval), as the framework sets this limit to reduce the damage from a malicious driver. We assume a bad driver is likely to continue this trend without encouragement to change his/her behaviour. Also, in our first scheme, the RSU reward and punishment mechanism does not consider any event and driver specific information while allocating the reward or punishment. This is straightforward but also somewhat simplistic. Thus, we then develop a more sophisticated RSU evaluation scheme which considers past driver behaviour, severity of an incident, and an RSU confidence score. We select a fuzzy logic approach as it can handle inexactness and incompleteness introduced at an RSU when resolving a dispute and to assign an appropriate amount of reward and punishment. The fixed and fuzzy RSU assessment schemes are compared to evaluate their performance and the results suggest that the fuzzy system is able to allocate a more justified amount of reward or punishment. This prolongs the lifespan of a driver by delaying access-blocking. The fuzzy model is more reasonable as it considers environmental dynamics (which changes from one incident to another incident, from one driver to another driver, and the severity of the event) whereas the fixed system does not consider any factor when allocating rewards of punishments. Despite this, the fuzzy system requires additional communication (between RSU and TA) and processing of the input to feed into the fuzzifier.

Initially, we use a simple probabilistic model for driver behaviour where the action of the driver is just based on a random distribution or, in some instances a configurable value. However, this does not capture the ability of “real” drivers to modify their behaviour based on past experience. The approach is memoryless. To address this limitation, we create a Markov chain driver behaviour model to control the driver behaviour from various trust states. It is expected that a more highly trusted driver will send

more trustworthy messages than a driver with a lower trust score. The driver behaviour model captures this behaviour by defining probabilistic distribution of true and untrue announcements in such a way that for each state the driver probabilistic behaviour is adjusted. This driver model is analysed in Veins by enabling announcements from all states, if possible. This is done by allocating different initial trust scores to drivers in different experiments. Analysis of the behaviour model confirms that the vehicle or driver which achieves trust continues announcing more trustworthy messages and remains trustworthy in the system. In contrast, vehicles / drivers which lose trust announce more untrue messages which result their eviction from the network. However, some drivers can announce both true and untrue messages which is also observed from both higher and lower trust states. Furthermore, the analysis confirms that reporters send less untrue announcements when their driver model is in a higher trust state.

In this research, a sender-side trust framework is presented which only needs a TPD to process the trust of all drivers individually. Thus, there is no need to disseminate any trust metrics for accepting or rejecting a message by receivers which reduces the performance of receiver-side trust systems. This strategy reduces communication overhead and driver decision time relating to an event.

The framework thwarts untrue attacks and inconsistent attacks (where a driver switches between trustworthy and untrue message announcements). The untrue detection mechanism only needs trust and opinion data to be collected at an RSU. The RSU immediately informs nearby RSUs to avoid invoking the costly dispute resolution process. The scheme avoids trust metrics dissemination as much as possible. If trust metrics dissemination is frequently allowed, it impacts communication which is unrelated to event announcements. Our framework requires trust data to be saved occasionally into a centralized TA server when a driver needs to change a vehicle. Incident information must also be stored for analysis and later verification. Disputes about events are not resolved locally by vehicles, rather an entity which belongs to the authority (i.e. an RSU) takes responsibility for deciding on disputes and the consequential actions. Using the fuzzy logic reward and punishment engine within the RSU, it does not dispense a fixed reward or punishment. Consideration is given to the severity of event and driver recent interactions with the network.

The driver behaviour model captures the likelihood that trustworthy drivers remain so, and vice versa. This is achieved through a Markov-chain state transition model. Furthermore, the magnitude of RSU rewards and punishments are such that a poor driver performing a “good deed” is rewarded less, than a similar highly trusted driver. The same philosophy is applied to highly trusted drivers that suddenly behave in a malicious manner. The punishment is relatively small to a highly trusted driver and the amount rises for similar future punitive actions.

7.2 Conclusion

Overall, a sender-side trust framework for VANETs with multiple types of vehicles, RSUs, and a TA is presented to avoid the significant burden of receiver-side evaluations of traffic events which saves decision time and communication overhead. Thus, receivers are relieved from costly communication and computation upon arrival of a traffic event message unless one of them notices the said event is not in place. Also, the framework leaves space to include many other classes of vehicle. A Tamper-Proof Device (TPD) inside all regular vehicles undertakes all the computing and management related to trust adjustment of drivers / vehicles based on their actions. This lowers the burden of processing at RSUs and communication required for trust adjustment of drivers and vehicles. Additionally, an untrue detection scheme detects and thwarts untrue, inconsistent, and cooperation attacks. A dispute resolution process can allocate either fixed or fuzzy rewards and punishments based on the decision of the dispute resolver RSU. The fuzzy scheme allocates rewards or punishments more appropriately considering driver past behaviour, severity of incident, and RSU confidence in the data. Driver announcements and reporting are also regulated based on their trust score using a Markov-chain driver behaviour model, and malicious or benevolent behaviour is appropriately recompensed using the fuzzy scheme of reward and punishment. Within this thesis, complex methodology is applied only when it is required to deal a problem. Hence, we have not used blockchain or machine learning approaches.

7.3 Future Work

The framework can be further studied to explore measures to thwart additional security threats. An improved security approach can be incorporated within the trust framework to address authentication, integrity, and nonrepudiation. Additionally, the privacy of the driver and/or vehicle can be further studied to discover an appropriate method to use with this framework. The performance of the trust framework can be compared with additional trust models to critically appraise the proposed scheme under different circumstances and confirm the claim that sender-side trust management models remain more efficient than the receiver-side schemes. This comparison could be based on other metrics besides the communication overhead and the response time. Currently, the RSU does not apply any filtering mechanism to detect malicious feedback which may bias accurate classification of untrue attacks. A suitable filtering scheme could be incorporated.

A machine learning approach could be investigated to employ at the sender-side to determine whether the driver's input to the system is valid based on the location and speed data. When an input is an outlier, the TPD could determine if a punishment should be issued to the driver. To this end, a machine learning-based outlier detection approach could be used to detect correlated multivariate outliers where a normalized dataset is not required. Supervised outlier detection techniques treat this as a classification problem which classifies data into outlier and inlier using some labelled data. For example, Support Vector Machine (SVM) can be used to classify new data based on the existing labelled

datasets. However, the timer buttons on the dashboard to record the event notice time and event location can avoid the implementation of this outlier detection with the trust framework. The parameters used for the TPD evaluation of announcements could be investigated further to determine if additional inputs to the system would be beneficial. Additionally, multivariate linear regression could be employed for estimating the reward or punishment for announcements rather than the current fixed rules. Also, the set of rules could be made adaptive according to different VANET scenarios. For example, in an urban scenario, the rules might be different from the rules used for highways. The framework already supports multiple drivers running a vehicle, but the research does not consider a single driver running multiple vehicles. For this latter case, the framework could employ a wireless card reader and card-based driver authentication mechanism with the TPD.

The framework is only based on two types of vehicle (i.e. regular and authorised), but in the real world, there are many vehicle classes on the road. For example, road maintenance vehicles, state-running buses, etc. These could be included. It is assumed the RSU infrastructure covers the entire road area. To reduce the overall cost it could be worth considering RSUs only having a limited, non-overlapping footprint. Currently, only a limited selection of traffic message announcements and service facility queries are supported by the framework. The list of services could be extended.

This research does not consider any fault handling mechanism associated with the TPD, vehicle dashboard, and RSU as well as any associated risk management. These elements could be explored. Furthermore, inputs to the fuzzy system related to an event and driver could be investigated further to improve the fuzzy RSU controller to better estimate the appropriate output values. For example, currently, all rules have equal weight during the fuzzy inference process; this might be improved by weight adjustment. Finally, the driver behaviour model could be examined with different probabilistic distributions for different trust states, or the Markov model could involve additional states that more accurately represent human behaviour.

The framework is scalable as long as infrastructure is available along roads and highways. To deploy this as a service to VANET users, we believe significant infrastructure needs to be developed. Truly, it has developed in some areas. For example, in London, there are some roadside displays which tell drivers to switch their engine off to keep the air quality at a good index when they are waiting at a signal. Also, large displays are used on highways to inform drivers that an accident has taken place at a road 1.5 miles ahead. So, drivers can take an early detour, if available. This message is not communicated in the way we mention in this thesis, but they are announced on roadside displays. However, we foresee they will be common along the roadsides to cover the entire road area. Some infrastructure is already deployed and autonomous cars and electric vehicles are manufactured with many built-in sensors and actuators helpful for VANET interaction. Additionally, the consideration of roadside infrastructure is common in most existing trust frameworks. The deployment of an authority

control and monitoring centre is in the jurisdiction of RTA. They can easily launch these sites at different places in a country to manage a VANET from an authoritative perspective. So, communication will not be disrupted due to the lack of infrastructure support. Additionally, we believe drivers should be trained from an ethical perspective of driving to develop appropriate behaviour in VANETs. For example, this course can deliver content on honesty, morality, situation awareness and understanding, and the advantage of sharing events. The content would also cover the negative consequences of sharing an untrue announcement in a VANET. Drivers should know that it is their responsibility to share road situations with other drivers. A driver who announces an event today helps others and he will receive announcements as help from others in the future.

Appendix A

A.1 Additional Traffic Scenarios

A.1.1 Debris on Road

A vehicle observing debris on a road broadcasts a message. RSU forwards this event directly to TA which in turn looks for official personnel from the RTA to sort out the problem. Once the debris on road "X" is sorted, then the TA sends a message saying "resolved debris on road" to RSU. RSU then retransmits this sorted message periodically multiple times within its range. Additionally, RSU sends this message to nearby RSU which also rebroadcasts the sorted message within its range. Hence, all vehicles within the range are notified and could use that road again. This sequence diagram for debris on road is shown in Fig. A.1-1. The process of broadcasting debris on a road is shown in Fig. A.1-2.

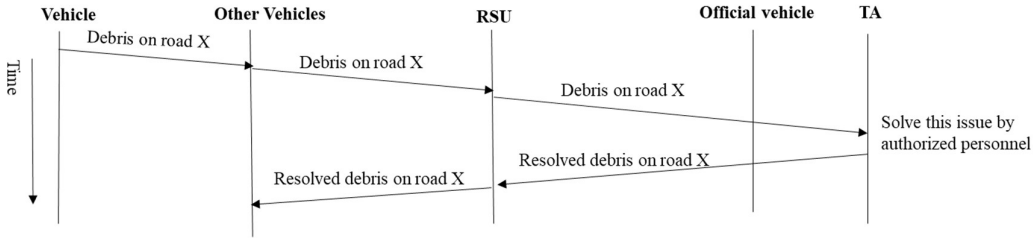


Fig. A.1-1: Sequence Diagram of Broadcasting a Debris on Road Message

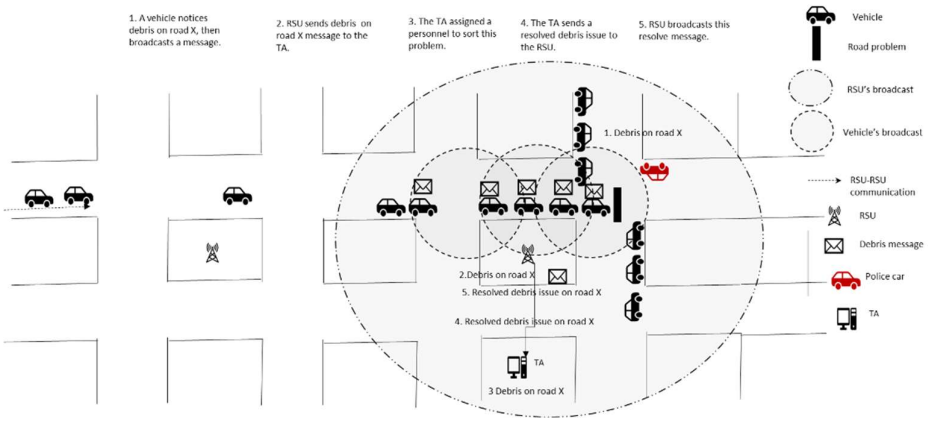


Fig. A.1-2: Process of Resolving a Debris on a Road

A.1.2 Announcement of Service Discovery from a Regular Vehicle

Suppose a vehicle is looking for a location of a nearby petrol pump and asking for help from an RSU by saying "any nearby petrol pump?". Meanwhile, this message reaches a nearby RSU by traversing intermediate vehicles. Then an RSU replies to this message by saying "the location of a nearby petrol pump is on road X". The vehicle uses this message as the source of the information as it is looking for it a moment ago. Fig. A.1-3 shows a sequence diagram of any service lookup in the

VANET and Fig. A.1-4 depicts a sequence diagram of a petrol pump locator in VANET. Fig. A.1-5 shows the process of finding a petrol pump in a VANET.

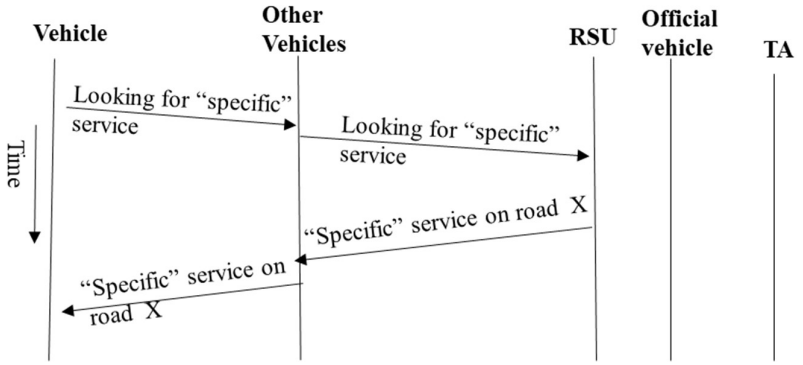


Fig. A.1-3: Sequence Diagram to Find a Service in a VANET

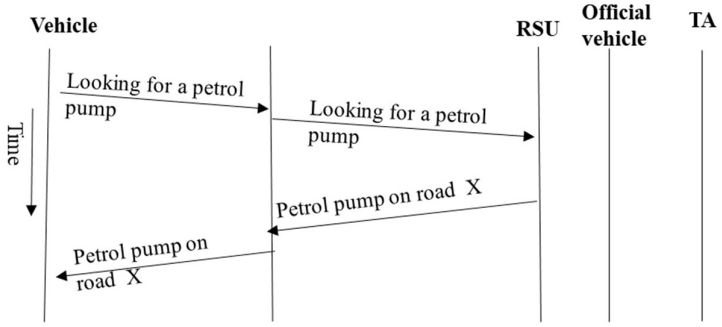


Fig. A.1-4: Sequence Diagram to Find a Petrol Pump in a VANET

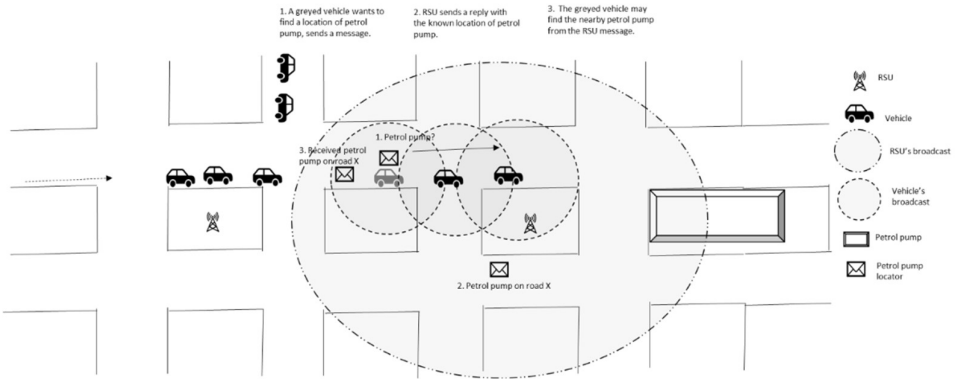


Fig. A.1-5: Process of Finding the Nearest Petrol Pump Station

A.1.3 Announcement of Road Defects

A vehicle when observing a road defect on a road broadcasts a message. RSU forwards this event directly to the TA which in turn looks for authoritative personnel to sort out the problem by contacting the RTA. Once the problem is sorted, the TA sends a message saying "road defect is sorted"

to RSU. RSU then broadcasts this sorted message periodically multiple times within its range. Additionally, RSU sends this message to nearby RSUs which also retransmit this message within their range. Hence, all the vehicles within range are notified about the sorted road defect and may use the road again. A sequence diagram to illustrate the operation sequence is shown in Fig. A.1-5. The process of solving road defects is shown in Fig. A.1-6.

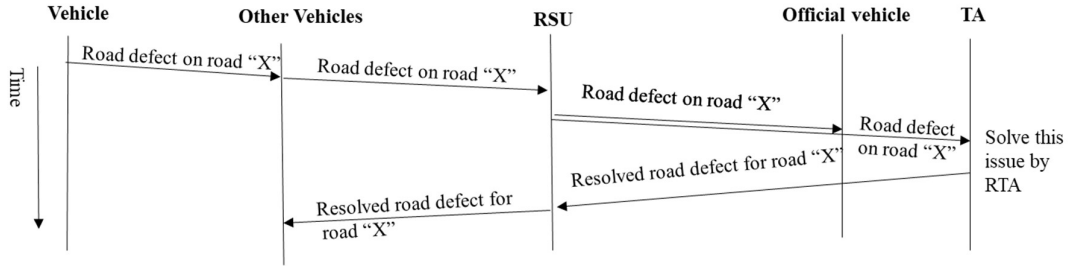


Fig. A.1-5: Sequence Diagram of Solving a Road Defect in VANET

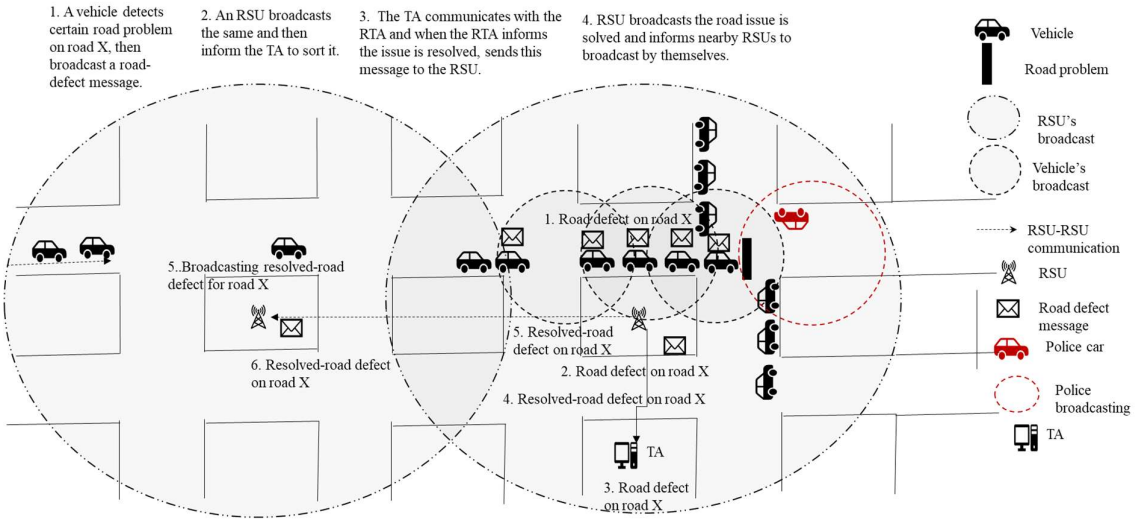


Fig. A.1-6: Process of Resolving a Road Defect in VANET

A.1.4 Announcement of Flooding

A vehicle when observing flood on a road broadcasts a message about this. When an RSU receives it, sends this event directly to the TA which in turn looks for the authoritative workforce to sort out the problem. Once the flood on road X is sorted, the TA sends a message saying "resolved flood on road" to RSU. RSU then broadcasts this sorted message periodically multiple times within its range. Additionally, RSU sends this message to nearby RSU which also retransmits the sorted message within their range. Hence, all the vehicles within range are notified about this event and may use the road again as it is free now. This sequence diagram is shown in Fig. A.1-7. The process of broadcasting flood on a road is shown in Fig. A.1-8.

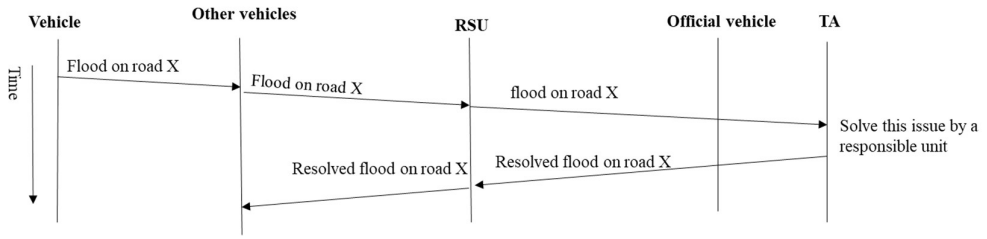


Fig. A.1-7: Sequence Diagram of Broadcasting Flooding on Road Message in VANET

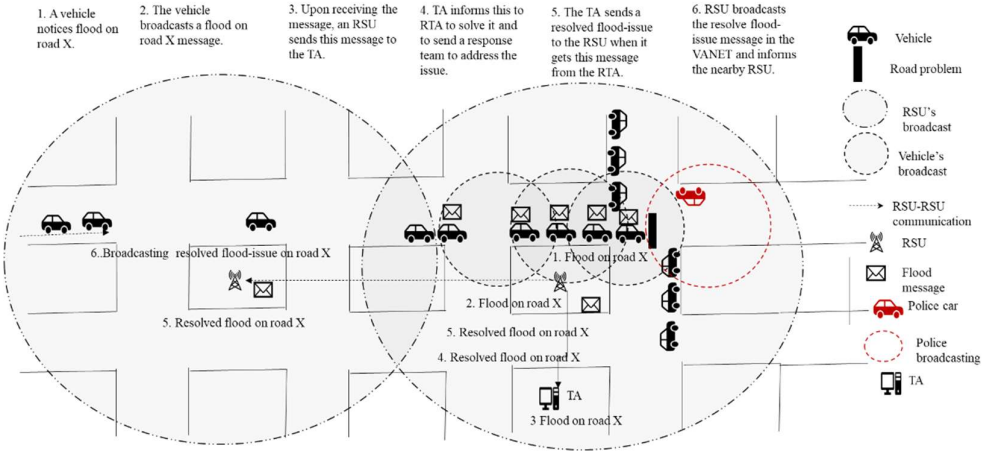


Fig. A.1-8: Process of Broadcasting Flood on Road Message on a Road in VANET

A.1.5 Announcement of Traffic Signal Malfunction

A vehicle when observing a traffic signal problem on road "X" broadcasts a message about this event. RSU sends this event directly to TA which in turn looks for authoritative personnel to sort out the problem on road. Once the traffic signal problem on road "X" is sorted, TA sends a message saying "resolved traffic signal problem on road X" to RSU. RSU then broadcasts this sorted message periodically multiple times within its range. Additionally, RSU sends this message to nearby RSU which also retransmits the sorted message within their range. Hence, all the vehicles within range are notified about traffic updates and may use the road again as it is free now. This sequence diagram for traffic signal malfunction event broadcasting is shown in Fig. A.1-9. The process of broadcasting a traffic signal problem is shown in Fig. A.1-10.

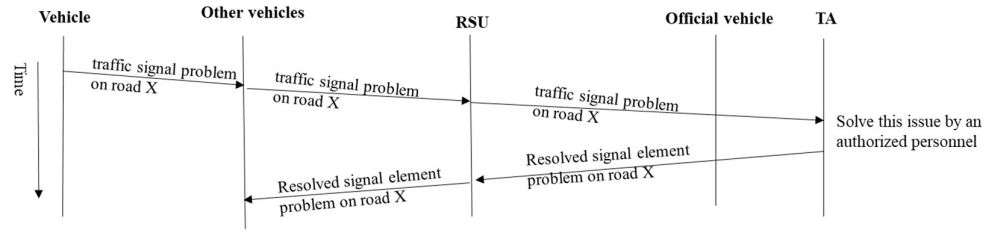


Fig. A.1-9: Sequence Diagram of Broadcasting a Traffic Signal Problem on a Road

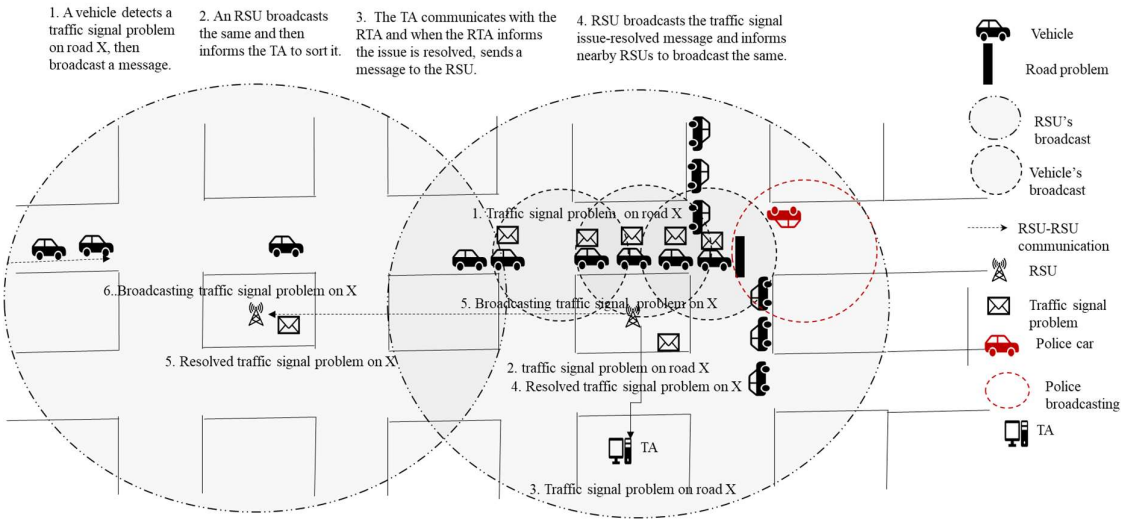


Fig. A.1-10: Process of Broadcasting a Traffic Signal Problem Message on a Road

References

[1] Jiang, D. and Delgrossi, L., 2008, May. IEEE 802.11 p: Towards an international standard for wireless access in vehicular environments. In *VTC Spring 2008-IEEE vehicular technology conference* (pp. 2036-2040). IEEE. doi:10.1109/VETECS.2008.458.

[2] Road accidents and safety statistics, Accessed: September 20, 2023. Available at: https://www.gov.uk/government/collections/road-accidents-and-safety-statistics (Accessed on 12 Dec. 23).

[3] How many vehicles are there in Great Britain? Accessed: June 8, 2023. Available at: https://www.racfoundation.org/motoring-faqs/mobility#a1(Accessed on 12 Dec. 23).

[4] Vehicles in use - Europe 2020. Accessed: June 8, 2023. Available at: https://www.acea.be/statistics/article/report-vehicles-in-use-europe-2020/ (Accessed on 12 Dec. 23).

[5] Number of vehicles in operation in the United States between 1st quarter 2018 and 1st quarter 2023. Accessed: June 8, 2023. Available at: https://www.statista.com/statistics/859950/vehicles-in-operation-by-quarter-united-states/ (Accessed on 12 Dec. 23).

[6] Lu, Z., Qu, G. and Liu, Z., 2018. A survey on recent advances in vehicular network security, trust, and privacy. *IEEE Transactions on Intelligent Transportation Systems*, *20*(2), pp.760-776. doi: 10.1109/TITS.2018.2818888.

[7] Tangade, S. and Manvi, S.S., 2017, August. Trust management scheme in VANET: Neighbour communication based approach. In *2017 International conference on smart technologies for smart nation (SmartTechCon)* (pp. 741-744). IEEE. doi: 10.1109/SmartTechCon.2017.8358469.

[8] Wei, Z., Yu, F.R. and Boukerche, A., 2014, September. Trust based security enhancements for vehicular ad hoc networks. In *Proceedings of the fourth ACM international symposium on Development and analysis of intelligent vehicular networks and applications* (pp. 103-109). doi: https://doi.org/10.1145/2656346.2656353.

[9] Tangade, S. and Manvi, S.S., 2018, November. CBTM: Cryptography based trust management scheme for secure vehicular communications. In *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)* (pp. 325-330). IEEE. doi: 10.1109/ICARCV.2018.8581173.

[10] Dahiya, R., Jiang, F. and Doss, R.R., 2020, December. A Feedback-Driven Lightweight Reputation Scheme for IoV. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)* (pp. 1060-1068). IEEE. doi: 10.1109/TrustCom50675.2020.00141.

[11] Ahmad, F., Franqueira, V.N. and Adnane, A., 2018. TEAM: A trust evaluation and management framework in context-enabled vehicular ad-hoc networks. *IEEE Access*, *6*, pp.28643-28660. doi: 10.1109/ACCESS.2018.2837887.

[12] Gazdar, T., Belghith, A. and Abutair, H., 2017. An enhanced distributed trust computing protocol for VANETs. *IEEE Access*, *6*, pp.380-392. doi: 10.1109/ACCESS.2017.2765303.

[13] Haddadou, N., Rachedi, A. and Ghamri-Doudane, Y., 2014. A job market signaling scheme for incentive and trust management in vehicular ad hoc networks. *IEEE Transactions on Vehicular Technology*, *64*(8), pp.3657-3674. doi: 10.1109/TVT.2014.2360883.

[14] Mühlbauer, R. and Kleinschmidt, J.H., 2018. Bring your own reputation: A feasible trust system for vehicular ad hoc networks. *Journal of Sensor and Actuator Networks*, *7*(3), p.37. doi: https://doi.org/10.3390/jsan7030037.

[15] Guleng, S., Wu, C., Chen, X., Wang, X., Yoshinaga, T. and Ji, Y., 2019. Decentralized trust evaluation in vehicular Internet of Things. *IEEE Access*, *7*, pp.15980-15988. doi: 10.1109/ACCESS.2019.2893262.

[16] Wei, Y.C. and Chen, Y.M., 2014, September. Adaptive decision making for improving trust establishment in VANET. In *The 16th Asia-Pacific Network Operations and Management Symposium* (pp. 1-4). IEEE. doi: 10.1109/APNOMS.2014.6996523.

[17] Sommer, C., Eckhoff, D., Brummer, A., Buse, D.S., Hagenauer, F., Joerer, S. and Segata, M., 2019. Veins: The open source vehicular network simulation framework. Recent Advances in Network Simulation: The OMNeT++ Environment and its Ecosystem, pp.215-252. doi: https://doi.org/10.1007/978-3-030-12842-5_6.

[18] Varga, A., 2010. OMNeT++. Modeling and tools for network simulation, pp.35-59. Available at: http://src.gnu-darwin.org/ports/science/omnetpp/work/omnetpp-2.3p1/doc/usman.pdf (Accessed on 12 Dec. 23).

[19] Hasrouny, H., Samhat, A.E., Bassil, C. and Laouiti, A., 2017. VANet security challenges and solutions: A survey. *Vehicular Communications*, *7*, pp.7-20. doi: https://doi.org/10.1016/j.vehcom.2017.01.002.

[20] Rostamzadeh, K., Nicanfar, H., Torabi, N., Gopalakrishnan, S. and Leung, V.C., 2015. A context-aware trust-based information dissemination framework for vehicular networks. *IEEE Internet of Things journal*, *2*(2), pp.121-132. doi: 10.1109/JIOT.2015.2388581.

[21] Pournaghi, S.M., Zahednejad, B., Bayat, M. and Farjami, Y., 2018. NECPPA: A novel and efficient conditional privacy-preserving authentication scheme for VANET. Computer Networks, 134, pp.78-92. doi: https://doi.org/10.1016/j.comnet.2018.01.015.

[22] Jing, T., Pei, Y., Zhang, B., Hu, C., Huo, Y., Li, H. and Lu, Y., 2018. An efficient anonymous batch authentication scheme based on priority and cooperation for VANETs. *EURASIP Journal on Wireless Communications and Networking*, *2018*(1), pp.1-13. doi: https://doi.org/10.1186/s13638-018-1294-z.

[23] Studer, A., Bai, F., Bellur, B. and Perrig, A., 2009. Flexible, extensible, and efficient VANET authentication. *Journal of Communications and Networks*, *11*(6), pp.574-588. doi: 10.1109/JCN.2009.6388411.

[24] Huang, J.L., Yeh, L.Y. and Chien, H.Y., 2010. ABAKA: An anonymous batch authenticated and key agreement scheme for value-added services in vehicular ad hoc networks. *IEEE Transactions on vehicular technology*, *60*(1), pp.248-262. doi: 10.1109/TVT.2010.2089544.

[25] Aquino-Santos, R., Rangel-Licea, V., García-Ruiz, M.A., González-Potes, A., Álvarez-Cardenas, O. and Edwards-Block, A., 2009. Inter-vehicular communications using wireless Ad Hoc networks. In *Automotive informatics and communicative systems: principles in vehicular networks and data exchange* (pp. 120-138). IGI Global. doi: 10.4018/978-1-60566-338-8.ch007

[26] Engoulou, R.G., Bellaïche, M., Pierre, S. and Quintero, A., 2014. VANET security surveys. *Computer Communications*, *44*, pp.1-13. doi: https://doi.org/10.1016/j.comcom.2014.02.020

[27] Arif, M., Wang, G., Bhuiyan, M.Z.A., Wang, T. and Chen, J., 2019. A survey on security attacks in VANETs: Communication, applications and challenges. *Vehicular Communications*, *19*, p.100179. doi: https://doi.org/10.1016/j.vehcom.2019.100179.

[28] Upadhyaya, A.N. and Shah, J.S., 2018. Attacks on vanet security. *Int J Comp Eng Tech*, *9*(1), pp.8-19.

[29] Zhou, T., Choudhury, R.R., Ning, P. and Chakrabarty, K., 2007, August. Privacy-preserving detection of sybil attacks in vehicular ad hoc networks. In *2007 Fourth Annual International*

Conference on Mobile and Ubiquitous Systems: Networking & Services (MobiQuitous) (pp. 1-8). IEEE. doi: 10.1109/MOBIQ.2007.4451013.

[30] Kerrache, C.A., Calafate, C.T., Cano, J.C., Lagraa, N. and Manzoni, P., 2016. Trust management for vehicular networks: An adversary-oriented overview. *IEEE Access*, *4*, pp.9293-9307. Doi: 10.1109/ACCESS.2016.2645452.

[31] Butler Jr, J.K., 1991. Toward understanding and measuring conditions of trust: Evolution of a conditions of trust inventory. *Journal of management*, *17*(3), pp.643-663. doi: https://doi.org/10.1177/014920639101700307.

[32] Mayer, R.C., Davis, J.H. and Schoorman, F.D., 1995. An integrative model of organizational trust. *Academy of management review*, *20*(3), pp.709-734. doi: https://doi.org/10.5465/amr.1995.9508080335.

[33] Kumar, A. and Sinha, M., 2014, March. Overview on vehicular ad hoc network and its security issues. In *2014 International conference on computing for sustainable global development (INDIACom)* (pp. 792-797). IEEE. doi: 10.1109/IndiaCom.2014.6828071

[34] Karnadi, F.K., Mo, Z.H. and Lan, K.C., 2007, March. Rapid generation of realistic mobility models for VANET. In *2007 IEEE wireless communications and networking conference* (pp. 2506-2511). IEEE. doi: 10.1109/WCNC.2007.467

[35] Samara, G., Al-Salihy, W.A. and Sures, R., 2010, May. Security issues and challenges of vehicular ad hoc networks (VANET). In *4th International Conference on New Trends in Information Science and Service Science* (pp. 393-398). IEEE.

[36] Dhurandher, S.K., Obaidat, M.S., Jaiswal, A., Tiwari, A. and Tyagi, A., 2010, December. Securing vehicular networks: a reputation and plausibility checks-based approach. In *2010 IEEE Globecom Workshops* (pp. 1550-1554). IEEE. doi: 10.1109/GLOCOMW.2010.5700199.

[37] Papadimitratos, P., Buttyan, L., Holczer, T., Schoch, E., Freudiger, J., Raya, M., Ma, Z., Kargl, F., Kung, A. and Hubaux, J.P., 2008. Secure vehicular communication systems: design and architecture. *IEEE Communications magazine*, *46*(11), pp.100-109. doi: 10.1109/MCOM.2008.4689252.

[38] Karimireddy, T. and Bakshi, A.G.A., 2016, March. A hybrid security framework for the vehicular communications in VANET. In *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)* (pp. 1929-1934). IEEE. doi: 10.1109/WiSPNET.2016.7566479.

[39] De Fuentes, J.M., González-Tablas, A.I. and Ribagorda, A., 2011. Overview of security issues in vehicular ad-hoc networks. In *Handbook of research on mobility and computing: Evolving technologies and ubiquitous impacts* (pp. 894-911). IGI global. doi: 10.4018/978-1-60960-042- 6.ch056.

[40] Chandra, S., Paira, S., Alam, S.S. and Sanyal, G., 2014, November. A comparative survey of symmetric and asymmetric key cryptography. In *2014 international conference on electronics, communication and computational engineering (ICECCE)* (pp. 83-93). IEEE. doi: 10.1109/ICECCE.2014.7086640.

[41] Li, J., Lu, H. and Guizani, M., 2014. ACPN: A novel authentication framework with conditional privacy-preservation and non-repudiation for VANETs. *IEEE Transactions on Parallel and Distributed Systems*, *26*(4), pp.938-948. doi: 10.1109/TPDS.2014.2308215.

[42] Sakhreliya, S.C. and Pandya, N.H., 2014, December. PKI-SC: Public key infrastructure using symmetric key cryptography for authentication in VANETs. In *2014 IEEE International Conference on Computational Intelligence and Computing Research* (pp. 1-6). IEEE. doi: 10.1109/ICCIC.2014.7238326.

[43] Shim, K.A., 2012. ${\cal CPAS} $: an efficient conditional privacy-preserving authentication scheme for vehicular sensor networks. *IEEE Transactions on Vehicular Technology*, *61*(4), pp.1874-1883. doi: 10.1109/TVT.2012.2186992.

[44] Bradai, A. and Afifi, H., 2011, May. A framework using IBC achieving non-repudiation and privacy in vehicular network. In *2011 Conference on Network and Information Systems Security* (pp. 1-6). IEEE. doi: 10.1109/SAR-SSI.2011.5931386.

[45] Bhavesh, N.B., Maity, S. and Hansdah, R.C., 2013, March. A protocol for authentication with multiple levels of anonymity (AMLA) in VANETs. In *2013 27th International Conference on Advanced Information Networking and Applications Workshops* (pp. 462-469). IEEE. doi: 10.1109/WAINA.2013.4.

[46] Wagan, A.A. and Jung, L.T., 2014, June. Security framework for low latency VANET applications. In *2014 international conference on computer and information sciences (ICCOINS)* (pp. 1-6). IEEE. doi: 10.1109/ICCOINS.2014.6868395.

[47] Al-Riyami, S.S. and Paterson, K.G., 2003, November. Certificateless public key cryptography. In *International conference on the theory and application of cryptology and information security* (pp. 452-473). Springer, Berlin, Heidelberg. doi: https://doi.org/10.1007/978-3-540- 40061-5_29.

[48] Sheikh, M.S., Liang, J. and Wang, W., 2019. A survey of security services, attacks, and applications for vehicular ad hoc networks (vanets). *Sensors*, *19*(16), p.3589. doi: https://doi.org/10.3390/s19163589.

[49] Horng, S.J., Tzeng, S.F., Huang, P.H., Wang, X., Li, T. and Khan, M.K., 2015. An efficient certificateless aggregate signature with conditional privacy-preserving for vehicular sensor networks. *Information Sciences*, *317*, pp.48-66. doi: https://doi.org/10.1016/j.ins.2015.04.033.

[50] Cui, J., Zhang, J., Zhong, H., Shi, R. and Xu, Y., 2018. An efficient certificateless aggregate signature without pairings for vehicular ad hoc networks. *Information Sciences*, *451*, pp.1-15. doi: https://doi.org/10.1016/j.ins.2018.03.060.

[51] Qu, F., Wu, Z., Wang, F.Y. and Cho, W., 2015. A security and privacy review of VANETs. *IEEE Transactions on Intelligent Transportation Systems*, *16*(6), pp.2985-2996. doi: 10.1109/TITS.2015.2439292.

[52] Sun, Y., Feng, Z., Hu, Q. and Su, J., 2012. An efficient distributed key management scheme for group-signature based anonymous authentication in VANET. *Security and Communication Networks*, *5*(1), pp.79-86. doi: https://doi.org/10.1002/sec.302.

[53] Agarwal, A. and Saraswat, R., 2013. A survey of group signature technique, its applications and attacks. *International Journal of Engineering and Innovative Technology (IJEIT)*, *2*(10), pp.28- 35.

[54] Malina, L., Castella-Roca, J., Vives-Guasch, A. and Hajny, J., 2012, October. Short-term linkable group signatures with categorized batch verification. In *International Symposium on Foundations and Practice of Security* (pp. 244-260). Springer, Berlin, Heidelberg. doi: https://doi.org/10.1007/978-3-642-37119-6_16.

[55] Hu, C., Chim, T.W., Yiu, S.M., Hui, L.C. and Li, V.O., 2012. Efficient HMAC-based secure communication for VANETs. *Computer Networks*, *56*(9), pp.2292-2303. doi: https://doi.org/10.1016/j.comnet.2012.04.002.

[56] Sun, J., Zhang, C., Zhang, Y. and Fang, Y., 2010. An identity-based security system for user privacy in vehicular ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, *21*(9), pp.1227-1239. doi: 10.1109/TPDS.2010.14.

[57] Lin, X., Lu, R., Liang, X. and Shen, X., 2011, April. STAP: A social-tier-assisted packet forwarding protocol for achieving receiver-location privacy preservation in VANETs. In *2011 Proceedings IEEE INFOCOM* (pp. 2147-2155). IEEE. doi: 10.1109/INFCOM.2011.5935026.

[58] Raya, M. and Hubaux, J.P., 2005, November. The security of vehicular ad hoc networks. In *Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks* (pp. 11-21). doi: https://doi.org/10.1145/1102219.1102223.

[59] Elmahdi, E. and Yoo, S.M., 2020, November. Secure data integrity in vanets based on CS-DC scheme. In *2020 IEEE Latin-American Conference on Communications (LATINCOM)* (pp. 1-5). IEEE. doi: 10.1109/LATINCOM50620.2020.9282267.

[60] Choi, J.Y., Jakobsson, M. and Wetzel, S., 2005, October. Balancing auditability and privacy in Vehicular networks. In *Proceedings of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks* (pp. 79-87). doi: https://doi.org/10.1145/1089761.1089775.

[61] Kaur, N. and Kad, S., 2016. A review on security related aspects in vehicular ad hoc networks. *Procedia computer science*, *78*, pp.387-394. doi: https://doi.org/10.1016/j.procs.2016.02.079.

[62] Jiang, S., Zhu, X. and Wang, L., 2016. An efficient anonymous batch authentication scheme based on HMAC for VANETs. *IEEE Transactions on Intelligent Transportation Systems*, *17*(8), pp.2193-2204. doi: 10.1109/TITS.2016.2517603.

[63] Zhang, L., Wu, Q., Domingo-Ferrer, J., Qin, B. and Hu, C., 2016. Distributed aggregate privacy- preserving authentication in VANETs. *IEEE Transactions on Intelligent Transportation Systems*, *18*(3), pp.516-526. doi: 10.1109/TITS.2016.2579162.

[64] Chim, T.W., Yiu, S.M., Hui, L.C. and Li, V.O., 2012. VSPN: VANET-based secure and privacy- preserving navigation. *IEEE Transactions on Computers*, *63*(2), pp.510-524. doi: 10.1109/TC.2012.188.

[65] Lu, R., Lin, X., Liang, X. and Shen, X., 2011. A dynamic privacy-preserving key management scheme for location-based services in VANETs. *IEEE Transactions on Intelligent Transportation Systems*, *13*(1), pp.127-139. doi: 10.1109/TITS.2011.2164068.

[66] Lin, X. and Li, X., 2013. Achieving efficient cooperative message authentication in vehicular ad hoc networks. *IEEE Transactions on Vehicular Technology*, *62*(7), pp.3339-3348. doi: 10.1109/TVT.2013.2257188.

[67] Chuang, M.C. and Lee, J.F., 2013. TEAM: Trust-extended authentication mechanism for vehicular ad hoc networks. *IEEE Systems Journal*, *8*(3), pp.749-758. doi: 10.1109/JSYST.2012.2231792.

[68] Shukla, K., Jha, C.K. and Shukla, A., 2014, April. How to Ensure the Availability of Communication Channel for Event Driven Message in VANET. In *2014 Fourth International Conference on Communication Systems and Network Technologies* (pp. 331-335). IEEE. doi: 10.1109/CSNT.2014.73.

[69] Cardote, A., Sargento, S. and Steenkiste, P., 2010, December. On the connection availability between relay nodes in a VANET. In *2010 IEEE Globecom Workshops* (pp. 181-185). IEEE. doi: 10.1109/GLOCOMW.2010.5700255.

[70] Kumar, S., Javaid, N., Yousuf, Z., Kumar, H., Khan, Z.A. and Qasim, U., 2012, October. On link availability probability of routing protocols for urban scenario in VANETs. In *2012 IEEE Conference on Open Systems* (pp. 1-6). IEEE. doi: *10.1109/ICOS.2012.6417632.*

[71] Johnson, D., Hu, Y.C. and Maltz, D., 2007. RFC 4728: The dynamic source routing protocol (DSR) for mobile ad hoc networks for IPv4. *Internet RFCs*.

[72] Perkins, C., Belding-Royer, E. and Das, S., 2003. RFC3561: Ad hoc on-demand distance vector (AODV) routing.

[73] Gerla, M., Hong, X. and Pei, G., Fisheye state routing protocol (FSR) for ad hoc networks. IETF Draft (2002).

[74] Ali, A.S. and Salem, F.M., 2017, December. A robust distributed authorization scheme for availability enhancement in VANET. In *2017 12th International Conference on Computer Engineering and Systems (ICCES)* (pp. 592-599). IEEE. doi: 10.1109/ICCES.2017.8275375.

[75] Okamoto, J. and Ishihara, S., 2010, December. Distributing location-dependent data in VANETs by guiding data traffic to high vehicle density areas. In *2010 IEEE Vehicular Networking Conference* (pp. 189-196). IEEE. doi: 10.1109/VNC.2010.5698248.

[76] Park, S. and Lee, S., 2012. Improving data accessibility in vehicle ad hoc network. *International Journal of Smart Home*, *6*(4), pp.169-176.

[77] Qian, Y. and Moayeri, N., 2008, May. Design of secure and application-oriented VANETs. In *VTC Spring 2008-IEEE Vehicular Technology Conference* (pp. 2794-2799). IEEE. doi: 10.1109/VETECS.2008.610.

[78] Moustafa, H., Bourdon, G. and Gourhant, Y., 2006, May. Providing authentication and access control in vehicular network environment. In *IFIP International Information Security Conference* (pp. 62-73). Springer, Boston, MA. doi: https://doi.org/10.1007/0-387-33406-8_6.

[79] Nakamoto, S., 2008. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, p.21260.

[80] Jiang, T., Fang, H. and Wang, H., 2018. Blockchain-based internet of vehicles: Distributed network architecture and performance analysis. *IEEE Internet of Things Journal*, *6*(3), pp.4640- 4649. doi: 10.1109/JIOT.2018.2874398.

[81] Chen, C., Wu, J., Lin, H., Chen, W. and Zheng, Z., 2019. A secure and efficient blockchain-based data trading approach for internet of vehicles. *IEEE Transactions on Vehicular Technology*, *68*(9), pp.9110-9121. doi: 10.1109/TVT.2019.2927533.

[82] Dwivedi, S.K., Amin, R., Das, A.K., Leung, M.T., Choo, K.K.R. and Vollala, S., 2022. Blockchain-based vehicular ad-hoc networks: A comprehensive survey. *Ad Hoc Networks*, p.102980. doi: https://doi.org/10.1016/j.adhoc.2022.102980.

[83] Yang, Z., Yang, K., Lei, L., Zheng, K. and Leung, V.C., 2018. Blockchain-based decentralized trust management in vehicular networks. *IEEE Internet of Things Journal*, *6*(2), pp.1495-1505. doi: 10.1109/JIOT.2018.2836144.

[84] Ahmed, S., Al-Rubeaai, S. and Tepe, K., 2017. Novel trust framework for vehicular networks. *IEEE Transactions on Vehicular Technology*, *66*(10), pp.9498-9511. doi: 10.1109/TVT.2017.2710124.

[85] Hu, H., Lu, R., Zhang, Z. and Shao, J., 2017. REPLACE: A reliable trust-based platoon service recommendation scheme in VANET. *IEEE Transactions on Vehicular Technology*, *66*(2), pp.1786-1797. doi: 10.1109/TVT.2016.2565001.

[86] Chim, T.W., Yiu, S.M., Yeung, C.Y., Li, V.O. and Hui, L.C., 2013, December. Secure, privacy- preserving, distributed motor vehicle event data recorder. In *2013 International Conference on* Connected Vehicles and Expo (ICCVE) (pp. 337-342). IEEE. doi: 10.1109/ICCVE.2013.6799817.

[87] Raya, M. and Hubaux, J.P., 2007. Securing vehicular ad hoc networks. *Journal of Computer Security*, *15*(1), pp.39-68. doi: 10.3233/JCS-2007-15103.

[88] Sumra, I.A. and Hasbullah, H.B., 2015, February. Using TPM to ensure security, trust and privacy (STP) in VANET. In *2015 5th National Symposium on Information Technology: Towards New Smart World (NSITNSW)* (pp. 1-6). IEEE. doi: 10.1109/NSITNSW.2015.7176402.

[89] Pearson, S. and Balacheff, B., 2003. *Trusted computing platforms: TCPA technology in context*. Prentice Hall Professional.

[90] Hussain, R., Lee, J. and Zeadally, S., 2020. Trust in VANET: A survey of current solutions and future research opportunities. *IEEE Transactions on Intelligent Transportation Systems*, *22*(5), pp.2553-2571. doi: 10.1109/TITS.2020.2973715.

[91] Huang, Z., Ruj, S., Cavenaghi, M.A., Stojmenovic, M. and Nayak, A., 2014. A social network approach to trust management in VANETs. *Peer-to-Peer Networking and Applications*, *7*(3), pp.229-242. doi: https://doi.org/10.1007/s12083-012-0136-8.

[92] Li, Q., Malip, A., Martin, K.M., Ng, S.L. and Zhang, J., 2012. A reputation-based announcement scheme for VANETs. IEEE Transactions on Vehicular Technology, 61(9), pp.4095-4108. Doi: 10.1109/TVT.2012.2209903.

[93] Li, X., Liu, J., Li, X. and Sun, W., 2013, September. RGTE: A reputation-based global trust establishment in VANETs. In *2013 5th International Conference on Intelligent Networking and Collaborative Systems* (pp. 210-214). IEEE. doi: 10.1109/INCoS.2013.91.

[94] Li, B., Liang, R., Zhu, D., Chen, W. and Lin, Q., 2020. Blockchain-based trust management model for location privacy preserving in VANET. *IEEE Transactions on Intelligent Transportation Systems*, *22*(6), pp.3765-3775. doi: 10.1109/TITS.2020.3035869.

[95] Mrabet, K., El Bouanani, F. and Ben-Azza, H., 2021, December. Dependable Decentralized Reputation Management System for Vehicular Ad Hoc Networks. In *2021 4th International Conference on Advanced Communication Technologies and Networking (CommNet)* (pp. 1-7). IEEE. doi: 10.1109/CommNet52204.2021.9641962.

[96] Pu, C., 2021, April. A novel blockchain-based trust management scheme for vehicular networks. In *2021 Wireless Telecommunications Symposium (WTS)* (pp. 1-6). IEEE. doi: 10.1109/WTS51064.2021.9433711.

[97] Chen, X., Ding, J. and Lu, Z., 2020. A decentralized trust management system for intelligent transportation environments. *IEEE Transactions on Intelligent Transportation Systems*, *23*(1), pp.558-571. doi: 10.1109/TITS.2020.3013279.

[98] Li, F., Guo, Z., Zhang, C., Li, W. and Wang, Y., 2021. ATM: an active-detection trust mechanism for VANETs based on blockchain. *IEEE Transactions on Vehicular Technology*, *70*(5), pp.4011-4021. doi: 10.1109/TVT.2021.3050007.

[99] Haddaji, A., Ayed, S. and Fourati, L.C., 2020, December. Blockchain-based Multi-Levels Trust Mechanism Against Sybil Attacks for Vehicular Networks. In *2020 IEEE 14th International Conference on Big Data Science and Engineering (BigDataSE)* (pp. 155-163). IEEE. doi: 10.1109/BigDataSE50710.2020.00028.

[100] Luo, B., Li, X., Weng, J., Guo, J. and Ma, J., 2019. Blockchain enabled trust-based location privacy protection scheme in VANET. *IEEE Transactions on Vehicular Technology*, *69*(2), pp.2034-2048. IEEE. doi: 10.1109/TVT.2019.2957744.

[101] Yang, Y.T., Chou, L.D., Tseng, C.W., Tseng, F.H. and Liu, C.C., 2019. Blockchain-Based Traffic Event Validation and Trust Verification for VANETs. *IEEE Access*, *7*, pp.30868-30877. IEEE. doi: 10.1109/ACCESS.2019.2903202.

[102] Xie, L., Ding, Y., Yang, H. and Wang, X., 2019. Blockchain-based secure and trustworthy Internet of Things in SDN-enabled 5G-VANETs. *IEEE Access*, *7*, pp.56656-56666. IEEE. doi: 10.1109/ACCESS.2019.2913682.

[103] Yang, Z., Wang, R., Wu, D., Yang, B. and Zhang, P., 2021. Blockchain-enabled trust management model for the Internet of Vehicles. *IEEE Internet of Things Journal*, *10(14),* pp. 12044 – 12054. IEEE. doi: 10.1109/JIOT.2021.3124073.

[104] Huang, F., Li, Q. and Zhao, J., 2022, August. Trust Management Model of VANETs Based on Machine Learning and Active Detection Technology. In *2022 IEEE/CIC International Conference on Communications in China (ICCC Workshops)* (pp. 412-416). IEEE. doi: 10.1109/ICCCWorkshops55477.2022.9896700.

[105] Malhi, A.K. and Batra, S., 2017. Fuzzy-based trust prediction for effective coordination in vehicular ad hoc networks. *International Journal of Communication Systems*, *30*(6), p.e3111. doi: https://doi.org/10.1002/dac.3111.

[106] Sharma, A. and Jaekel, A., 2021, July. Machine learning approach for detecting location spoofing in VANET. In *2021 International Conference on Computer Communications and Networks (ICCCN)* (pp. 1-6). IEEE. doi: 10.1109/ICCCN52240.2021.9522170.

[107] Mankodiya, H., Obaidat, M.S., Gupta, R. and Tanwar, S., 2021, October. XAI-AV: Explainable Artificial Intelligence for Trust Management in Autonomous Vehicles. In *2021 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)* (pp. 1-5). IEEE. doi: 10.1109/CCCI52664.2021.9583190.

[108] Tangade, S., Manvi, S.S. and Hassan, S., 2019, September. A deep learning-based driver classification and trust computation in VANETs. In *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)* (pp. 1-6). IEEE. doi: 10.1109/VTCFall.2019.8891462.

[109] Zhang, D., Yu, F.R. and Yang, R., 2018, December. A machine learning approach for software- defined vehicular ad hoc networks with trust management. In *2018 IEEE Global Communications Conference (GLOBECOM)* (pp. 1-6). IEEE. doi: 10.1109/GLOCOM.2018.8647426.

[110] Guo, J., Li, X., Liu, Z., Ma, J., Yang, C., Zhang, J. and Wu, D., 2020. TROVE: A context-awareness trust model for VANETs using reinforcement learning. *IEEE Internet of Things Journal*, *7*(7), pp.6647-6662. Doi: 10.1109/JIOT.2020.2975084.

[111] Shaikh, R.A. and Alzahrani, A.S., 2014. Intrusion-aware trust model for vehicular ad hoc networks. *Security and Communication Networks*, *7*(11), pp.1652-1669. doi: https://doi.org/10.1002/sec.862.

[112] Wu, A., Ma, J. and Zhang, S., 2011, September. RATE: a RSU-aided scheme for data-centric trust establishment in VANETs. In *2011 7th International Conference on Wireless Communications, Networking and Mobile Computing* (pp. 1-6). IEEE. doi: 10.1109/wicom.2011.6040302.

[113] Wei, Y.C. and Chen, Y.M., 2012, June. An efficient trust management system for balancing the safety and location privacy in VANETs. In *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications* (pp. 393-400). IEEE. doi: 10.1109/TrustCom.2012.79.

[114] Najafi, M., Khoukhi, L. and Lemercier, M., 2021, October. A Multidimensional Trust Model for Vehicular Ad-Hoc Networks. In 2021 IEEE 46th Conference on Local Computer Networks (LCN) (pp. 419-422). IEEE. doi: 10.1109/LCN52139.2021.9524960.

[115] Mármol, F.G. and Pérez, G.M., 2012. TRIP, a trust and reputation infrastructure-based proposal for vehicular ad hoc networks. *Journal of Network and Computer Applications*, *35*(3), pp.934- 941. doi: https://doi.org/10.1016/j.jnca.2011.03.028.

[116] Zhou, A., Li, J., Sun, Q., Fan, C., Lei, T. and Yang, F., 2015. A security authentication method based on trust evaluation in VANETs. EURASIP Journal on Wireless Communications and Networking, 2015(1), pp.1-8. doi: ttps://doi.org/10.1186/s13638-015-0257-x.

[117] Dotzer, F., Fischer, L. and Magiera, P., 2005, June. Vars: A vehicle ad-hoc network reputation system. In *Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks* (pp. 454-456). IEEE. doi: 10.1109/WOWMOM.2005.109.

[118] Verma, N. and Jain, A., 2017, April. Trust management in mobile ad hoc network: A survey. In *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)* (Vol. 2, pp. 455-460). IEEE. doi: 10.1109/ICECA.2017.8212856.

[119] Abdelwahab, S., Gaber, T. and Wahed, M., 2017. Trust-based security models in wireless sensor networks: a survey. *International Journal of Computational Intelligence Studies*, *6*(2-3), pp.245-266. doi: https://doi.org/10.1504/IJCISTUDIES.2017.089057.

[120] Siddiqui, S.A., Mahmood, A., Sheng, Q.Z., Suzuki, H. and Ni, W., 2021, October. A Time-aware Trust Management Heuristic for the Internet of Vehicles. In 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) (pp. 1-8). IEEE. doi: 10.1109/TrustCom53373.2021.00019.

[121] Uma, E., Senthilnayaki, B., Devi, A., Rajeswary, C. and Dharanyadevi, P., 2021, December. Trust Score Evaluation Scheme for Secure Routing in VANET. In *2021 IEEE International Conference on Mobile Networks and Wireless Communications (ICMNWC)* (pp. 1-6). IEEE. doi: 10.1109/ICMNWC52512.2021.9688475.

[122] The Network Simulator-2 (NS-2). Available at: https://www.isi.edu/nsnam/ns/index.html (Accessed: 13 Dec. 23).

[123] Zhang, J., Zheng, K., Zhang, D. and Yan, B., 2020. AATMS: An anti-attack trust management scheme in VANET. *IEEE Access*, *8*, pp.21077-21090. Doi: 10.1109/ACCESS.2020.2966747.

[124] Saraswat, D. and Chaurasia, B.K., 2013, September. AHP based trust model in VANETs. In 2013 5th International Conference and Computational Intelligence and Communication Networks (pp. 391-393). IEEE. doi: 10.1109/CICN.2013.86.

[125] Kerrache, C.A., Lagraa, N., Calafate, C.T. and Lakas, A., 2017. TFDD: A trust-based framework for reliable data delivery and DoS defense in VANETs. *Vehicular Communications*, *9*, pp.254-267. doi: https://doi.org/10.1016/j.vehcom.2016.11.010.

[126] Atwa, R.J., Flocchini, P. and Nayak, A., 2021. A Fog-based Reputation Evaluation Model for VANETs. In *2021 International Symposium on Networks, Computers and Communications (ISNCC)* (pp. 1-7). IEEE. doi: 10.1109/ISNCC52172.2021.9615820.

[127] Abassi, R., Douss, A.B.C. and Sauveron, D., 2020. TSME: a trust-based security scheme for message exchange in vehicular Ad hoc networks. *Human-centric Computing and Information Sciences*, *10*(1), pp.1-19. doi: https://doi.org/10.1186/s13673-020-00248-4.

[128] Awan, K.A., Din, I.U., Almogren, A., Guizani, M. and Khan, S., 2020. StabTrust—A stable and centralized trust-based clustering mechanism for IoT enabled vehicular ad-hoc networks. *IEEE Access*, *8*, pp.21159-21177. doi: 10.1109/ACCESS.2020.2968948.

[129] Tangade, S., Manvi, S.S. and Lorenz, P., 2020. Trust management scheme based on hybrid cryptography for secure communications in VANETs. *IEEE Transactions on Vehicular Technology*, *69*(5), pp.5232-5243. doi: 10.1109/TVT.2020.2981127.

[130] Soleymani, S.A., Abdullah, A.H., Zareei, M., Anisi, M.H., Vargas-Rosales, C., Khan, M.K. and Goudarzi, S., 2017. A secure trust model based on fuzzy logic in vehicular ad hoc networks with fog computing. *IEEE Access*, *5*, pp.15619-15629. doi: 10.1109/ACCESS.2017.2733225.

[131] Chang, X., 1999, December. Network simulations with OPNET. In *Proceedings of the 31st conference on Winter simulation: Simulation---a bridge to the future-Volume 1* (pp. 307-314).

[132] van der Heijden, R.W., Lukaseder, T. and Kargl, F., 2018. VeReMi: A dataset for comparable evaluation of misbehavior detection in vanets. In *Security and Privacy in Communication Networks: 14th International Conference, SecureComm 2018, Singapore, Singapore, August 8- 10, 2018, Proceedings, Part I* (pp. 318-337). Springer International Publishing. doi: https://doi.org/10.1007/978-3-030-01701-9_18.

[133] Gurung, S., Lin, D., Squicciarini, A. and Bertino, E., 2013, June. Information-oriented trustworthiness evaluation in vehicular ad-hoc networks. In *International Conference on*

Network and System Security (pp. 94-108). Springer, Berlin, Heidelberg. doi: https://doi.org/10.1007/978-3-642-38631-2_8.

[134] Basheer, H.S., Bassil, C. and Chebaro, B., 2015, October. Toward using data trust model in VANETs. In *2015 International Conference on Applied Research in Computer Science and* Engineering (ICAR) (pp. 1-2). IEEE. doi: 10.1109/ARCSE.2015.7338136.

[135] Wan, J., Gu, X., Wang, J. and Chen, L., 2019. A Trust Scheme Based on Vehicles Reports of Events in VANETs. *Wireless Personal Communications*, *105*(1), pp.121-143. doi: https://doi.org/10.1007/s11277-018-6106-6.

[136] Rawat, D.B., Yan, G., Bista, B.B. and Weigle, M.C., 2015. Trust On the Security of Wireless Vehicular Ad-hoc Networking. *Ad Hoc & Sensor Wireless Networks*, *24*(3-4), pp.283-305.

[137] Wang, G. and Wu, Y., 2014, September. BIBRM: A Bayesian inference based road message trust model in vehicular ad hoc networks. In *2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications* (pp. 481-486). IEEE. doi: 10.1109/TrustCom.2014.137.

[138] Chen, Y.M. and Wei, Y.C., 2013. A beacon-based trust management system for enhancing user centric location privacy in VANETs. *Journal of Communications and Networks*, *15*(2), pp.153- 163. doi: 10.1109/JCN.2013.000028.

[139] Sharma, K. and Chaurasia, B.K., 2015, April. Trust based location finding mechanism in VANET using DST. In *2015 Fifth International Conference on Communication Systems and Network Technologies* (pp. 763-766). IEEE. doi: 10.1109/CSNT.2015.160.

[140] Peterson, L. Sunay, O. and Davie, B., 2023, Private 5G: A Systems Approach, Systems Approach, LLC.

[141] Crypto++ library. Available at: https://cryptopp.com. (Accessed on: 13 Dec. 23).

[142] Ahmed, S. and Tepe, K., 2016, April. Misbehaviour detection in vehicular networks using logistic trust. In *2016 IEEE Wireless Communications and Networking Conference* (pp. 1-6). IEEE. doi: 10.1109/WCNC.2016.7564966.

[143] Minhas, U.F., Zhang, J., Tran, T. and Cohen, R., 2010. Towards expanded trust management for agents in vehicular ad-hoc networks. *International Journal of Computational Intelligence: Theory and Practice (IJCITP)*, *5*(1), pp.03-15.

[144] Yao, X., Zhang, X., Ning, H. and Li, P., 2017. Using trust model to ensure reliable data acquisition in VANETs. *Ad Hoc Networks*, *55*, pp.107-118. doi: https://doi.org/10.1016/j.adhoc.2016.10.011.

[145] Atwa, R.J., Flocchini, P. and Nayak, A., 2020, October. Risk-based trust evaluation model for VANETs. In *2020 International Symposium on Networks, Computers and Communications (ISNCC)* (pp. 1-6). IEEE. doi: 10.1109/ISNCC49221.2020.9297329.

[146] Gao, H., Liu, C., Yin, Y., Xu, Y. and Li, Y., 2021. A hybrid approach to trust node assessment and management for vanets cooperative data communication: Historical interaction perspective. *IEEE Transactions on Intelligent Transportation Systems*, *23*(9), pp.16504-16513. doi: 10.1109/TITS.2021.3129458.

[147] Abdelaziz, K.C., Lagraa, N. and Lakas, A., 2014, August. Trust model with delayed verification for message relay in VANETs. In *2014 International Wireless Communications and Mobile Computing Conference (IWCMC)* (pp. 700-705). IEEE. doi: 10.1109/IWCMC.2014.6906441.

[148] Rehman, M.U., Ahmed, S., Khan, S.U., Begum, S. and Ishtiaq, A., 2018, March. ARV2V: Attack resistant vehicle to vehicle algorithm, performance in term of end-to-end delay and trust computation error in VANETs. In *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)* (pp. 1-6). IEEE. doi: 10.1109/ICOMET.2018.8346338.

[149] Pham, T.N.D. and Yeo, C.K., 2018. Adaptive trust and privacy management framework for vehicular networks. *Vehicular Communications*, *13*, pp.1-12. doi: https://doi.org/10.1016/j.vehcom.2018.04.006.

[150] Li, W. and Song, H., 2015. ART: An attack-resistant trust management scheme for securing vehicular ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, *17*(4), pp.960-969. doi: 10.1109/TITS.2015.2494017.

[151] Rai, I.A., Shaikh, R.A. and Hassan, S.R., 2020. A hybrid dual-mode trust management scheme for vehicular networks. *International Journal of Distributed Sensor Networks*, *16*(7), p.1550147720939372. doi: https://doi.org/10.1177/1550147720939372.

[152] Liu, Z., Weng, J., Ma, J., Guo, J., Feng, B., Jiang, Z. and Wei, K., 2019. TCEMD: A trust cascading-based emergency message dissemination model in VANETs. *IEEE Internet of Things Journal*, *7*(5), pp.4028-4048. doi: 10.1109/JIOT.2019.2957520.

[153] Koirala, B., Tangade, S.S. and Manvi, S.S., 2018, September. Trust management based on node stay time in VANET. In *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (pp. 242-248). IEEE. doi: 10.1109/ICACCI.2018.8554563.

[154] The Network Simulator (NS-3). Available at: www.nsnam.org, (Accessed on: 14 Dec. 23).

[155] Rehman, A., Hassan, M.F., Hooi, Y.K., Qureshi, M.A., Shukla, S., Susanto, E., Rubab, S. and Abdel-Aty, A.H., 2022. CTMF: Context-Aware Trust Management Framework for Internet of Vehicles. *IEEE Access*, *10*, pp.73685-73701. doi: 10.1109/ACCESS.2022.3189349.

[156] Hussain, R., Nawaz, W., Lee, J., Son, J. and Seo, J.T., 2016, August. A hybrid trust management framework for vehicular social networks. In *International Conference on Computational Social Networks* (pp. 214-225). Springer, Cham. doi: https://doi.org/10.1007/978-3-319-42345-6_19.

[157] Kerrache, C.A., Lagraa, N., Calafate, C.T., Cano, J.C. and Manzoni, P., 2016. T-VNets: A novel trust architecture for vehicular networks using the standardized messaging services of ETSI ITS. *Computer Communications*, *93*, pp.68-83. doi: https://doi.org/10.1016/j.comcom.2016.05.013.

[158] Arshad, M., Ullah, Z., Khalid, M., Ahmad, N., Khalid, W., Shahwar, D. and Cao, Y., 2018. Beacon trust management system and fake data detection in vehicular ad-hoc networks. *IET Intelligent Transport Systems*, *13*(5), pp.780-788. Doi: https://doi.org/10.1049/ietits.2018.5117.

[159] Dinesh, S. and Sonal, G., 2014. Qualnet simulator. *International Journal of Information & Computation Technology, ISSN*, pp.0974-2239.

[160] Rathi, A.K. and Santiago, A.J., 1990. The new NETSIM simulation program. *Traffic engineering & control*, *31*(5).

[161] Behrisch, M., Bieker, L., Erdmann, J. and Krajzewicz, D., 2011. SUMO–simulation of urban mobility: an overview. In Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation. ThinkMind.

[162] Lan, K.C., 2010. MOVE: a practical simulator for mobility model in VANET. In *Telematics communication technologies and vehicular networks: wireless architectures and applications* (pp. 355-368). IGI Global. doi: 10.4018/978-1-60566-840-6.ch021.

[163] Härri, J., Filali, F., Bonnet, C. and Fiore, M., 2006, September. VanetMobiSim: generating realistic mobility patterns for VANETs. In *Proceedings of the 3rd international workshop on Vehicular ad hoc networks* (pp. 96-97). doi: https://doi.org/10.1145/1161064.1161084.

[164] Zeng, X., Bagrodia, R. and Gerla, M., 1998, July. GloMoSim: a library for parallel simulation of large-scale wireless networks. In *Proceedings of the twelfth workshop on Parallel and distributed simulation* (pp. 154-161).

[165] Bennett, J., 2010. *OpenStreetMap*. Packt Publishing Ltd.

[166] Wegener, A., Piórkowski, M., Raya, M., Hellbrück, H., Fischer, S. and Hubaux, J.P., 2008, April. TraCI: an interface for coupling road traffic and network simulators. In *Proceedings of the 11th communications and networking simulation symposium* (pp. 155-163). doi: https://doi.org/10.1145/1400713.1400740.

[167] Piorkowski, M., Raya, M., Lugo, A.L., Papadimitratos, P., Grossglauser, M. and Hubaux, J.P., 2008. TraNS: realistic joint traffic and network simulator for VANETs. *ACM SIGMOBILE mobile computing and communications review*, *12*(1), pp.31-33. Doi: https://doi.org/10.1145/1374512.1374522.

[168] Wang, S.Y. and Chou, C.L., 2009. Nctuns simulator for wireless vehicular ad hoc network research. *Ad Hoc Networks: New Research*, pp.97-123

[169] Gazdar, T., Rachedi, A., Benslimane, A. and Belghith, A., 2012, December. A distributed advanced analytical trust model for VANETs. In *2012 IEEE Global Communications Conference (GLOBECOM)* (pp. 201-206). IEEE. doi: 10.1109/GLOCOM.2012.6503113.

[170] Goli-Bidgoli, S. and Movahhedinia, N., 2017. Determining vehicles' radio transmission range for increasing cognitive radio VANET (CR-VANET) reliability using a trust management system. *Computer Networks*, *127*, pp.340-351. Doi: https://doi.org/10.1016/j.comnet.2017.07.017.

[171] Liu, H., Han, D. and Li, D., 2021. Behavior analysis and blockchain based trust management in vanets. *Journal of Parallel and Distributed Computing*, *151*, pp.61-69. Doi: https://doi.org/10.1016/j.jpdc.2021.02.011.

[172] Agrawal, S., Raw, R.S., Tyagi, N. and Misra, A.K., 2015. Fuzzy logic based greedy routing (FLGR) in multi-hop vehicular ad hoc networks. *Indian Journal of Science and Technology*, *8*(30), pp.1-14. doi: 10.17485/ijst/2015/v8i30/70085.

[173] Zhou, Y., Li, H., Shi, C., Lu, N. and Cheng, N., 2018. A fuzzy-rule based data delivery scheme in VANETs with intelligent speed prediction and relay selection. *Wireless Communications and Mobile Computing*, *2018*. doi: https://doi.org/10.1155/2018/7637059.

[174] Igried, B., Alsarhan, A., Al-Khawaldeh, I., AL-Qerem, A. and Aldweesh, A., 2022. A Novel Fuzzy Logic-Based Scheme for Malicious Node Eviction in a Vehicular Ad Hoc Network. *Electronics*, *11*(17), p.2741. doi: https://doi.org/10.3390/electronics11172741.

[175] Inedjaren, Y., Zeddini, B., Maachaoui, M. and Barbot, J.P., 2019, November. Securing intelligent communications on the vehicular AdHoc networks using fuzzy logic based trust OLSR. In *2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA)* (pp. 1-6). IEEE. doi: 10.1109/AICCSA47632.2019.9035241.

[176] Hasan, M.M., Jahan, M., Kabir, S. and Wagner, C., 2021, July. A Fuzzy Logic-Based Trust Estimation in Edge-Enabled Vehicular Ad Hoc Networks. In *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (pp. 1-8). IEEE. doi: 10.1109/FUZZ45933.2021.9494428.

[177] Gayathri, M. and Gomathy, C., 2022, June. Fuzzy based Trusted Communication in Vehicular Ad hoc Network. In *2022 2nd International Conference on Intelligent Technologies (CONIT)* (pp. 1-4). IEEE. doi: 10.1109/CONIT55038.2022.9847823.

[178] Xia, H., San-shun, Z., Ben-xia, L., Li, L. and Xiang-guo, C., 2018. Towards a novel trust-based multicast routing for VANETs. *Security and Communication Networks*, *2018*. Doi: https://doi.org/10.1155/2018/7608198.

[179] Soleymani, S.A., Goudarzi, S., Anisi, M.H., Kama, N., Adli Ismail, S., Azmi, A., Zareei, M. and Hanan Abdullah, A., 2020. A trust model using edge nodes and a cuckoo filter for securing VANET under the NLoS condition. *Symmetry*, *12*(4), p.609. doi: https://doi.org/10.3390/sym12040609.

[180] Malhi, A.K. and Batra, S., 2017. Fuzzy-based trust prediction for effective coordination in vehicular ad hoc networks. *International Journal of Communication Systems*, *30*(6), p.e3111. doi: https://doi.org/10.1002/dac.3111.