Towards Security by Design of Connected and Automated Vehicles: Cyber and Physical Threats, Mitigations, and Architectures

by

Dajiang Suo

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2021

Author . . . .
Department of Mechanical Engineering
December 15, 2020

Certified by . . . .
Sanjay E. Sarma
Professor of Mechanical Engineering
Thesis Supervisor

Accepted by . . . .
Nicolas G. Hadjiconstantinou
Chairman, Department Committee on Graduate Theses

Towards Security by Design of Connected and Automated Vehicles: Cyber and Physical Threats, Mitigations, and Architectures

by

Dajiang Suo

Submitted to the Department of Mechanical Engineering
on December 15, 2020, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Mechanical Engineering

Abstract

Security, safety and privacy converge when it comes to the design of cyber-physical systems (CPS) such as connected and automated vehicles (CAVs). This trend can be attributed to the increased level of connectivity and automation and the new potential of insider attacks caused by changes in vehicle ownership. For example, A CAV whose on-board sensors, such as Light detection and ranging (LIDAR) and camera, are under spoofing attacks or subject to variations in environmental conditions (e.g., light, weather) may conduct risky maneuvers. Additionally, a CAV that can communicate with nearby vehicles, cloud servers, and roadside infrastructure can be turned into a “cyber-weapon” by adversaries to compromise transportation services or customer privacy. Designing mitigation solutions is a challenging task for Original equipment manufacturers who need to prioritize among safety, security, and privacy, and deal with ever-changing attack surfaces and the power of attackers.

This thesis proposes a security by design framework for identifying and mitigating cyber and physical threats on CAVs. A structured security engineering process for threat identification is first presented, which provides guidance to designing defensive mechanisms such that any compromise in design goals is traceable to a specific cyber or physical attack. After prioritizing among different identified threats, this thesis focuses on solutions to mitigate two types of threats: Physical threats on perception tasks with optical sensors and cyber threats on traffic event forgery in Vehicle-to-Infrastructure (V2I) communication.

Second, to mitigate physical threats to on-board optical sensors caused by environmental hazards, this thesis develops a object-recognition method based on light polarization. The proposed approach can provide multimodal data providing clues about the surface of objects, which complements the depth and RGB information from existing optical sensors. A proof-of-concept platform built in a laboratory benchtop verifies and evaluates the proposed concept.

Third, a secure V2I communication protocol titled “Proof-of-Travel” (POT) is developed to verify the authenticity of V2I messages. This novel approach utilizes and combines the physical laws of vehicle movement with cryptography mechanisms used for ensuring the security of distributed networks.

By developing and demonstrating these two proof-of-concept mitigation solutions, this thesis illustrates that security and safety goals for cyber-physical systems can be achieved more cost-effectively by following the security by design framework.

Thesis Supervisor: Sanjay E. Sarma
Title: Professor of Mechanical Engineering

Acknowledgments

First, I would like to thank my supervisor, Professor Sanjay Sarma, for believing in me and supporting me throughout my Ph.D. program such that I can survive. I am also deeply grateful for the help and guidance my committee members Prof. John Leonard and Prof. Jinhua Zhao. I would like to thank Dylan, Josh, and Pranay for helping me find my home at MIT Auto-ID lab . To other members in MIT Auto-ID lab, I would like to thank you all for your help and kindness that makes 35-206 an awesome place to work. As Sanjay always says, and I quote, "we are family."

Also, I want to give my special thanks to my colleagues, including Kyle, John, Matt and Sarah, at Ford Motor Company. Thank you all for believing in a "stranger" and supporting me during my Ph.D. life, especially during my intern in MI, which makes Dearborn my second hometown in the U.S. Also, to all my friends at MIT-CHIEF and Harvard, thanks for all the smiles, dinners, and happy times we spent together. Finally, I would like to thank my families, especially my mom, for giving me life, loving me and raising me up. I love you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Security, safety and privacy properties converge when it comes to the design of cyber-physical systems (CPS) such as connected and automated vehicles (CAVs). This can be attributed to the increased level of connectivity and automation and the potential of inside attacks due to changes in vehicle ownership. For example, A CAV whose onboard sensors such as Light detection and ranging (Lidar) or camera are under spoofing attacks or variations in environmental conditions (e.g., light, weather) may conduct risky maneuvers (e.g., unintended deceleration). Furthermore, a CAV that can communicate with nearby vehicles, cloud servers, or roadside infrastructure through vehicle-to-everything (V2X) communication can also be turned into a "cyber-weapon" by adversaries to compromise transportation services or customer privacy. Designing mitigation solutions is challenging not only because of the heterogeneous nature of each type of threat but also because of engineering trade-offs necessary for achieving security, safety, and privacy goals at the same time.

1.1 Motivation

There are several rising trends in security and safety engineering for developing CAV systems. These trends also reflect the urgent research needs for ensuring security, safety and privacy simultaneously with adopting new technologies and business models in the automotive sector. These changes are guided by technology development in original equipment providers (OEMs), new mobility-sharing services, and industry best practices and standards. Related standards include safety-related standards such as ISO 26262 [53] and Safety of the Intended Functionality (SOTIF) [51], security-related standards including SAE J3061 [116], ISO 21434 [52], and even privacy regulations such as General Data Protection Regulation (GDPR) [145], which is a European-wide privacy law that protects personal data (i.e., Personal Identification Information in the U.S.) for Europeans but can also affect OEMs in the U.S. if the vehicle is built and sold to the EU vehicle market.

- Safety and cybersecurity (including privacy) converge. One challenge of ensuring cybersecurity when designing fully automated vehicles is that safety-critical functions depend on external data sources such as on-board sensors, V2X communication, and remote services from cloud servers. Therefore, compromises in any of these channels can result in abnormal operations of intended functions [51] or losses of data assets, which calls for a unified engineering process. This trend is reflected in the fact that multiple OEMs have security experts collaborate with other CAV development teams by integrating cybersecurity into the system safety engineering process [53, 116].

- Changes in the levels of automation, connectivity, and car ownership create new attack surfaces for CAVs. There exist vehicles with 1-3 automation level (defined by SAE [28]) target at the market of private-owned passenger vehicles and are installed with enlarged sensor coverage [138, 5]. On the other hand, the enhanced connectivity and autonomy in CAVs become enablers for new ways of mobility-sharing services. Multiple OEMs have so far made the announcement to develop highly automated vehicles to support ride-sharing and goods delivery [29]. These vehicles are designed to support level 4-5 automation and often involve enhanced connectivity among individual vehicles, cloud centers for fleet operations, and Apps for ride requests, which create rich attack surfaces for adversaries [32].

- The role of digital infrastructure in ensuring the cybersecurity and safety of

CAVs has been evolving. A highly automated vehicle design must operate safely without relying on external information (e.g., from V2X or cloud servers through cellular networks) [96]. From an OEM's point of view, this requires efforts in designing robust and secure sensing and perception systems such that automated controllers will not misclassify objects or conduct risky maneuvers due to signal noise or malicious signal injection. The modules for vehicle decision-making and controls must tolerate risk conditions and guide CAVs out of dangerous zones even under failures or attacks. This design rationale is recommended in the discussion of minimum risk conditions in ISO 26262 [53]. At the same time, enhanced connectivity such as V2X communication channels with low latency, high bandwidth, and increased computing power also provide safety and efficiency benefits [89]. Individual automated vehicles can benefit from communication among vehicles and with roadside infrastructure by collaborating in perception and driving-related tasks. Transportation agencies and law enforcement departments can also benefit from real-time traffic information shared by every vehicle for enhanced agility in responding to emergencies. Therefore, it is in the interest of multiple transportation system stakeholders to mitigate cyber attacks. While the algorithms for V2X misbehavior detection can rely on feedback provided by on-board sensors, they can also utilize information provided by infrastructure components. In this paper, we present cyber threats and discuss mitigation solutions from both the vehicle-based and the infrastructure-based perspectives [54, 32].

1.2 Thesis contributions

Given the knowledge gap between arising cyber and physical threats on CAVs and the lack of engineering tools that support the identification and mitigation of these threats in a systematic manner, this thesis contributes to the following aspects:

1. A structured security engineering framework with software toolset for threat identification-Specifically, a concept model for threat modeling helps security,

safety, and CAV design teams within an OEM to work jointly for identifying cyber and physical threats that can cause safety hazards or compromise the new mobility services based on CAVs. Additionally, the concept model is implemented with architecture modeling language for complex systems-SysmL [44], which allows us to develop a virtualization to support the security engineering process.

2. A proof-of-concept design for the safety of on-board sensors-This thesis illustrates how to design a more robust optical sensor for object recognition tasks by leveraging the polarization property of light. This design can mitigate safety hazards on optical sensors due to extreme environment conditions near CAVs, such as low illumination conditions due to darkness.

3. A proof-of-concept design for the security of V2X communication-This thesis proposes a protocol for the secure communication between vehicles and infrastructure (V2I), titled Proof-of-Travel (POT). The POT protocol combines the physical law of vehicle movement with cryptography mechanisms. Although built upon cryptography techniques, including public-key infrastructure, digital signatures, and hash functions, POT achieves the goal of mitigating V2I misbehavior by transforming the power of cryptography into social and economic mechanisms to increase the cost of being malicious such that rational adversaries motivated by profit seeking will lose interest.

1.3 Thesis outline

Fig 1-1 shows the organization of the thesis. Chapter 2 reviews previous works on threat (hazard) identification and mitigation. Chapter 3 presents a security engineering framework including a new concept model for threat identification. The usage of the framework is illustrated through security analyses on CAVs for both private use and shared mobility services. Chapter 4 illustrates the design of optical sensors for improving CAV safety under adverse environmental conditions. In particular, a novel

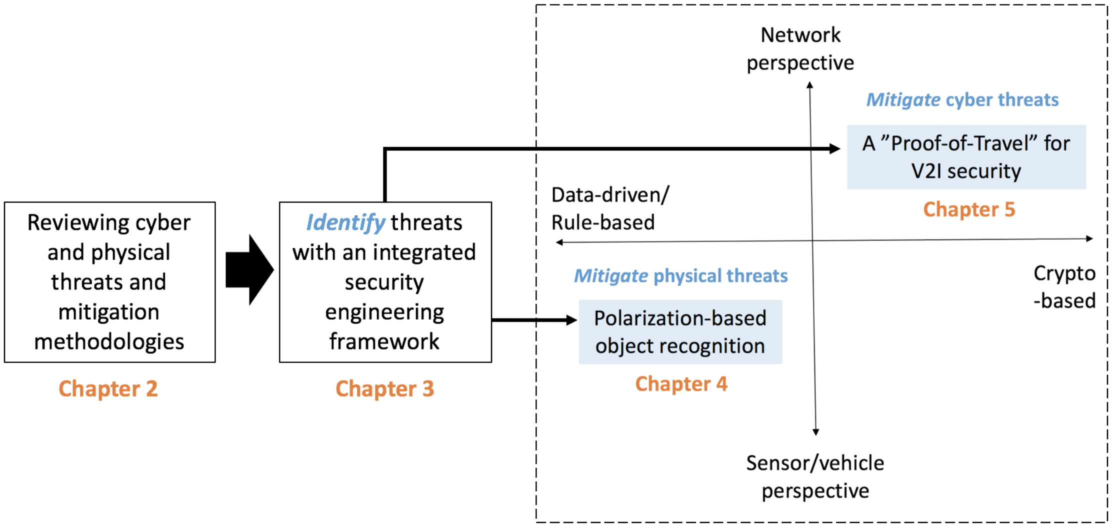

Reviewing cyber and physical threats and mitigation methodologies
Chapter 2
Identify threats with an integrated security engineering framework
Chapter 3
Network perspective
Mitigate cyber threats
A "Proof-of-Travel" for V2I security
Chapter 5
Data-driven/ Rule-based
Crypto -based
Mitigate physical threats
Polarization-based object recognition
Chapter 4
Sensor/vehicle perspective

Figure 1-1: The organization of this thesis

method for object recognition based on light polarization is developed and evaluated through benchtop experiments. In Chapter 5, vehicle-based and infrastructure-based approaches are proposed to mitigate malicious V2I messages. For the infrastructure-based approach, a Proof-of-Travel protocol (POT) that builds on vehicle and infrastructure communication is discussed and applied to evaluating the trustworthiness of V2I event reports.

Chapter 2

A Review of the Threat Landscape for Connected and Automated Vehicles

A review that surveys previous works on cyber threats and mitigation solutions for CAVs provides insights into knowledge gaps and new design challenges posed by new trends discussed in Chapter 1. Table 2.1 summarizes of existing literature that survey cyber and physical threats on CAVs.

To the best of our knowledge, most papers researching the cybersecurity issues of CAVs mainly focus on known threats discovered by the ethical-hacker community and the corresponding defenses for these identified threats [96, 101, 34, 152, 31, 109], as shown in Table. 2.1 There are also works that focus on a specific subsystem or aspect of CAV, including machine learning security [106], in-vehicle networks [61], human factors [69], or V2X communication [144, 117, 45, 3, 113, 157, 92, 42]. A comprehensive list of potential cyber attacks would be valuable to engineers who design mitigation solutions. However, none of these previous works can fully resolve the challenges mentioned in Chapter 1, and expand below.

First, none of these works illustrate a security engineering process engineers can follow to identify cyber and physical threats in a systematic way. In particular, the security engineering process developed in this thesis ensures that each identified threat

Table 2.1: Existing surveys on the cybersecurity of connected and automated vehicles-I

Authors	**Focus**	**Design goals**	**Automation levels**	**Connectivity**	**System components**
Parkinson et al. [96]	Knowledge gaps between known threats and existing mitigation	Security, safety, privacy	All levels	Physical and wireless access to in-vehicle networks, in-vehicle or wireless sensors, GPS, V2X, Wi-fi, cellular	CAVs, drivers, road infrastructure, cloud servers, pedestrians
Sheehan et al. [122]	Risk assesement and cyber-risk reduction by using Bayesian networks	Security, safety	Not mentioned explicitly, but vehicles with human drivers	The same as in [96], with a focus on GPS for CAV's navigation systems.	CAVs, drivers
Petit et al. [101]	A summary of cyber attacks on CAVs, their prioritization, and attack feasibility	Security, safety	Level 4-5	Physical and wireless access to in-vehicle networks, in-vehicle or wireless sensors, GPS, V2X, OBUs for digital maps	CAVs, drivers, road infrastructure, security credential management for V2X
Petit [98]	A stakeholder analysis on Avs and the ecosystem support their production and operation	Security, safety, privacy	Level 4-5	The same as in [101]	CAVs, drivers, road infrastructure, connected service providers, road contract, and fleet operators

Table 2.2: Existing surveys on the cybersecurity of connected and automated vehicles-II

Authors	**Focus**	**Design goals**	**Automation levels**	**Connectivity**	**System components**
Takahashi et al. [134]	A summary of cyber attacks in-vehicle networks, backend systems for connected services, and the communication between them	Security	Not mentioned	Physical and wireless access to in-vehicle networks, V2X, telematics	Connected vehicles, backend systems for connected services
Tokody et al. [141]	A review of security and safety co-engineering process based on industry standards	Security, safety	Level 4-5	Wireless sensors, V2X	CAVs, smart signage, smart lights, ITS traffic controller
Chattopadhyay et al. [21]	A brief summary of the use of security by design framework in designing cyber-physical systems	Security, safety, privacy	Not mentioned, but AVs without human drivers	In-vehicle wireless connection through bluetooth and smartphone, V2X, telematics	CAVs, road infrastructure
Eiza et al. [34]	A general discussion on cyber attacks through malware, OBD, or apps and mitigation solutions	Security, safety	Not mentioned	Physical access to ECUs through OBD ports, apps, or malware	Connected vehicles, backend systems for connected services

Table 2.3: Existing surveys on the cybersecurity of connected and automated vehicles-III

Authors	Focus	Design goals	Automation levels	Connectivity	System components
Yeh et al. [152]	A discussion on jamming, spoofing, and interference attacks on automotive radar and V2X modules	Security, safety, privacy	Not mentioned	On-board radar, V2X	CAVs and road infrastructure
Cui et al. [31]	A review of safety hazards and cyber threats	Security, safety	both CAVs with and without drivers	Physical access to in-vehicle networks through CAN, wireless sensors, V2X	CAVs and road infrastructure
Koscher et al. [61]	An experimental security analysis of ECUs and CAN bus	Security, safety	Not mentioned but vehicles with drivers	CAN, OBD-port, ECUs	regular vehicles
Qayyum et al. [106]	A review of machine learning security in the context of CAVs	Security, safety, privacy	Not mentioned	On-board wireless sensors, V2X	CAVs and road infrastructure

Table 2.4: Existing surveys on the cybersecurity of connected and automated vehicles-IV

Authors	**Focus**	**Design goals**	**Automation levels**	**Connectivity**	**System components**
Linkov et al. [69]	A review of human factors issues related to cyersecurity of AVs	Security, safety, privacy	Not mentioned explicitly but Vehicles with drivers	Not the focus	AVs and human drivers
Heijden et al. [144]	A review of malicious behaviors of vehicular networks	Security, safety	Not mentioned explicitly but Vehicles without drivers	V2X modules	CAVs
Ren et al. [109]	A review of cyber threats on V2X and solutions	Security, safety, privacy	both CAVs with and without drivers	Physical and wireless access to in-vehicle networks, on-board sensors	AVs

will be mitigated by at least one mitigation solution, establishing engineering traceability. While understanding security strengths and weaknesses of each mitigation solution is important, engineers also need to evaluate the undesired consequences and risks of specific threats based on each stakeholder's needs, such as crashes, leakage of passenger locations, or customer complaints due to delays in transportation services, etc.

Second, previous works have not have paid enough attention to resolving potential design conflicts. This is necessary for future smart transportation systems that are designed to meet multiple goals by multi-stakeholders such as security, privacy, and availability of service. Heijden et al. provide a taxonomy for misbehavior detection mechanisms and review the pros and cons of different schemes [144]. They describe the conflict between security and privacy resulting from the use of pseudonyms. Petit et al. describe similar conflict in a survey for pseudonym schemes of vehicular network [100]. However, these papers do not generalize the discussion to give readers a holistic view of resolving conflicts among multiple system properties. The derivation of cyber and physical threats with Attack Trees in this thesis provides help in this aspect.

Third, many recommendations on mitigation solutions in previous works are based on a static attack model, that is, the power of an adversary does not change during CAVs' life cycle, which is unrealistic. One example is the V2X communications in vehicular networks where adversaries can use stolen vehicle credentials to create Sybil attacks on VANET [75]. Stealing multiple vehicle credential is difficult and requires expertise in the early stage of CAV deployment but might become easier with increasing penetration of V2X technologies and with increasing number of OBUs deployed increases. A direct consequence is that an adversary is able to spoof multiple vehicle identities (Sybil nodes) by presenting valid credentials. This thesis addresses this issue by proposing a Proof-of-Travel protocol for determining the trustworthiness and reputation of vehicle nodes based on its interactions with infrastructure components.

Chapter 3

An Integrated Security Engineering Process for Threat Identification

To resolve the first challenge in CAV development, we present a security engineering process for identifying and organizing threats and adversarial behaviors, as shown in Fig. 3-1. The process includes 4 tasks: defining design goals, developing system architecture, documenting assumptions on attacker models, and deriving attack trees for organizing cyber and physical threats.

The framework aims to merge security (privacy) and safety analyses and is extended from the state-of-the-art work on security and safety engineering activities both in academia [132, 153, 79, 67] and the automotive industry [52, 51, 53, 116, 104, 133]. It enables joint work by multiple engineering teams and ensures that any undesired consequence which compromises design goals is traceable to a specific threat.

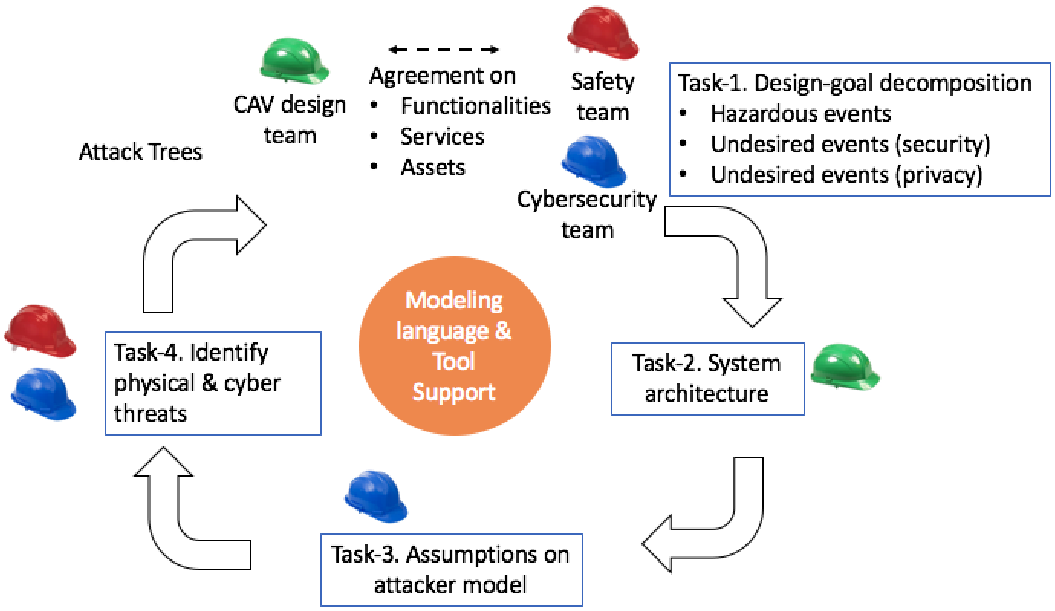

Attack Trees
CAV design team
Agreement on
• Functionalities
• Services
• Assets
Safety team
Cybersecurity team
Task-1. Design-goal decomposition
• Hazardous events
• Undesired events (security)
• Undesired events (privacy)
Modeling language & Tool Support
Task-2. System architecture
Task-3. Assumptions on attacker model
Task-4. Identify physical & cyber threats

Figure 3-1: The proposed security engineering process for threat identification. Three teams-safety, security and CAV design-need to first agree on key functionalities, services, and assets (e.g., PII or vehicle data) that are critical for the system operation. They then define hazardous or undesired events that compromise any of these design goals. The CAV design team defines a system architecture that includes CAV components, human principals that can interact with CAVs such as drivers, passengers, customers, or remote operators. At the same time, the security team documents assumptions made on the power and motivation of attackers. Based on the system architecture and attacker model, the security and safety teams, along with domain experts, identify and derive threats for each system component. The analysis results will be documented in the form of Attack Trees, which establish the mapping between high-level design goals and specific threats. The traceability provided by Attack Trees can ensure mitigation solutions are complete [2].

3.1 Design goal decomposition: Safety, security, and privacy

The process starts with three teams: safety (represented as a red hat), security (a blue hat), and CAV design (a green hat) teams agreeing on key functionalities, services, and assets, as shown in Fig. 3-1. They are high-level design goals that must be achieved for the normal operation of vehicles and CAV-enabled transportation services.

The vehicle ownership and the target market segmentations determine what types of automated-driving functions and connectivity a CAV needs to support. Based on trends in the automotive domain discussed earlier, we assume that CAVs with lower levels of automation (level 1-3 as defined by SAE [28]) and connectivity mainly serve the market of private passenger vehicles (hereinafter referred to as "Type-1 vehicles"), while level 4-5 automated vehicles with enhanced connectivity (e.g., V2X) target mobility-sharing services (hereinafter referred to as "Type-2 vehicles").

This categorization (Type-1 vs. Type-2) is not the only way to organize the security analysis but reflects a realistic organizational structure across different departments within certain vehicle manufactures [30]. For example, an OEM may assign a set of teams to the development of advanced driver-assistance systems (ADAS) that involve interaction with and intervention by human drivers, while a newly established division can focus on developing CAVs with fully automated-driving systems (ADS) to support ride-sharing services.

The high-level goals defined are then decomposed into more detailed hazardous events (HEs) for safety or undesired events (UEs) for security activities, which is also specified in automotive safety and security standards [53].

- CAV-related functionalities. For a Type-1 vehicle, it is expected to support the functions of warning human drivers about potential collision risks. On the other hand, a Type-2 vehicle must provide functions of automated braking, acceleration, or steering after a collision risk is detected. In addition, a Type-2 vehicle may also support remote control by fleet operators [68, 36].

- CAV-related services. A Type-1 CAV often provides customers with basic connected services such as remote health monitoring and roadside assistance through cellular networks. A Type-2 CAV takes services one step further to support ride-sharing and even become the enabler for intelligent traffic control and management. In that case, the traffic controller, public transportation agencies, law enforcement departments will rely on the information the CAV provides for allocating resources.

- CAV-related assets. We focus our discussion on digital assets (i.e., mobility data) generated by humans or vehicles. For a Type-1 vehicle, Personal Identifiable Information (PII) or location-based information stored within the vehicle or in a backend cloud server are assets and need security protection. The same holds for a Type-2 vehicle except location-based data might also be generated by new sensing or V2X modules.

Type-1 vehicle

We define three HEs regarding CAV functionalities and two UEs related to customer data.

HE-1 Unintended transition to human-control mode without proper warnings

HE-2 False alarms or false notifications to drivers

HE-3 Unintended maneuvers of CAVs

UE-1 Unauthorized access to driver/vehicle generated data

UE-2 Unauthorized tracking of vehicle movement

Type-2 vehicle

For a Type-2 CAV, we have different safety and security concerns and thus define different HEs and UEs. For example, a Type-2 CAV might violate traffic rules if the automated-driving system misclassifies traffic signs, which is hazardous (HE-4). In addition to the UE identified for Type-1, we can define new UEs if the CAV is used for fulfilling ride requests in ride-sharing services (UE-3) or as a sensing node providing real-time traffic and road conditions to the traffic management center for emergency responses to incidents and road hazards (UE-4).

HE-3 Unintended maneuvers of CAVs

HE-4 Violation of traffic rules/regulations

UE-1 Unauthorized access to driver/vehicle generated data

UE-2 Unauthorized tracking of vehicle movement

UE-3 CAVs do not respond to ride requests

UE-4 Transportation agencies or law enforcement does not respond to emergencies in time

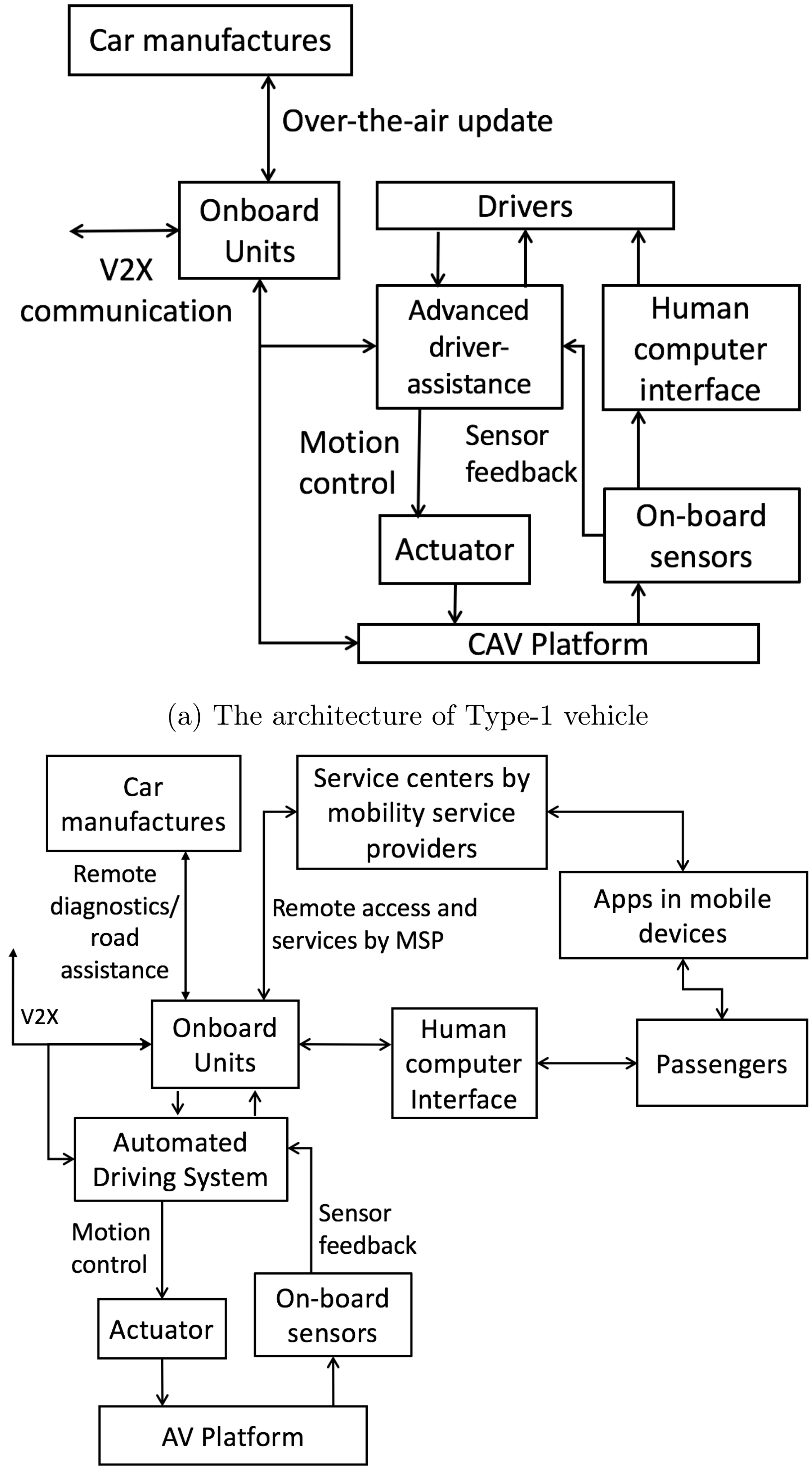
Part of these security goals and objectives are rooted in the traditional security objectives used in information systems such as the CIA (confidentiality, integrity, and availability) triad [118]. However, security goals for CAVs can have a broader scope for CAVs, including protecting digital assets such as customer and vehicle-generated data can be related to the confidentiality objective, protecting CAV key functionalities calls for message integrity, and the CAV-enable services must be protected to ensure the availability.

3.2 System architecture development

After identifying design goals for CAVs, the second task is to derive system architectures. A CAV's security architecture often includes system components, communication channels, and principals that interact with the system. Similar to design goals, the CAV architecture is determined by assumptions engineers make on vehicle autonomy, connectivity, and car ownership. Two candidate system architectures are presented to be consistent with the categorization of CAVs (Type-1 vs. Type-2) we made earlier, as shown in Fig. 3-2.

Fig. 3-2a represents the architecture of a Type-1 CAV, while Fig. 3-2b corresponds to a Type-2 CAV. We address differences between them in terms of autonomy, human-vehicle interactions, the storage of data, and physical and wireless access points by different human principals.

To begin, the most fundamental difference between Type-1 and Type-2 CAVs lies in vehicle autonomy or control authority. Both the ADAS (Fig. 3-2a) in auto-pilot mode and the ADS (Fig. 3-2b) rely on information from on-board sensors and wireless communication to determine when to brake, accelerate, or steer automatically. However, for a Type-1 CAV, the human driver is expected to monitor driving conditions and vehicle status and take over during abnormal situations, such as failures in the electronic, mechanical or power systems, when the vehicle is out of the operational design domain [53], or when the ADAS cannot determine the type of unknown objects.



(a) The architecture of Type-1 vehicle

(b) The architecture of Type-2 vehicle

Figure 3-2: Architectures for Type-1 and 2 connected and automated vehicles.

Therefore, the human-vehicle interface designed for telematics, entertainment applications, and driver notifications in Type-1 needs to be re-designed for Type-2 architectures. Both human-driver and human-passenger interactions reflect this need for redesign. To design the human-vehicle interface in Type-1 architectures, engineers need to account for the possibility that drivers must take over when cyber-attacks occurs. The interface needs to provide warnings to drivers about potential risks detected by sensors or indicated by V2X messages. For Type-2, the focus of interface design is passenger authentication through on-board interfaces or user Apps [73].

Another difference lies in the physical and wireless access points that have different attack surfaces. For example, in addition to the remote diagnostics and roadside assistance services that are often provided by car manufactures for Type-1 CAVs [4, 93] based on 3G or LTE cellular network, Type-2 CAVs often introduce two new types of remote connections, as shown in Fig. 3-2b. The first remote connection is the communication link between CAVs and the remote control center by the mobility service provider (MSP). The controller in the cloud center, either a human or an algorithm, will dispatch CAVs after receiving ride requests according to the status of each vehicle. The second remote connection is the high-speed link that supports teleoperation of moving CAVs out of gridlock in emergency situations [36, 68].

Different types of car ownership also lead to different processes and locations for data storage. For private vehicles, the collection, storage, and sharing of PII or sensor data are governed by "terms of service" that owners sign with OEMs when they subscribe to connected-vehicle services [131]. For level 4-5 CAVs providing mobility-sharing services, it is still unclear who will own vehicle-generated or perception data if vehicle status or driving conditions are transmitted to the cloud server for fleet management and ride dispatch, as shown in Fig. 3-2b.

Another difference involves on-board modules or wireless links that support V2X communication. Even though both types of CAVs use V2X modules, they support different functionalities. In addition to broadcasting vehicle movement and status to support safety-critical applications in Type-1, V2X modules installed in Type-2 vehicles can report traffic events with high-criticality to local infrastructure or the

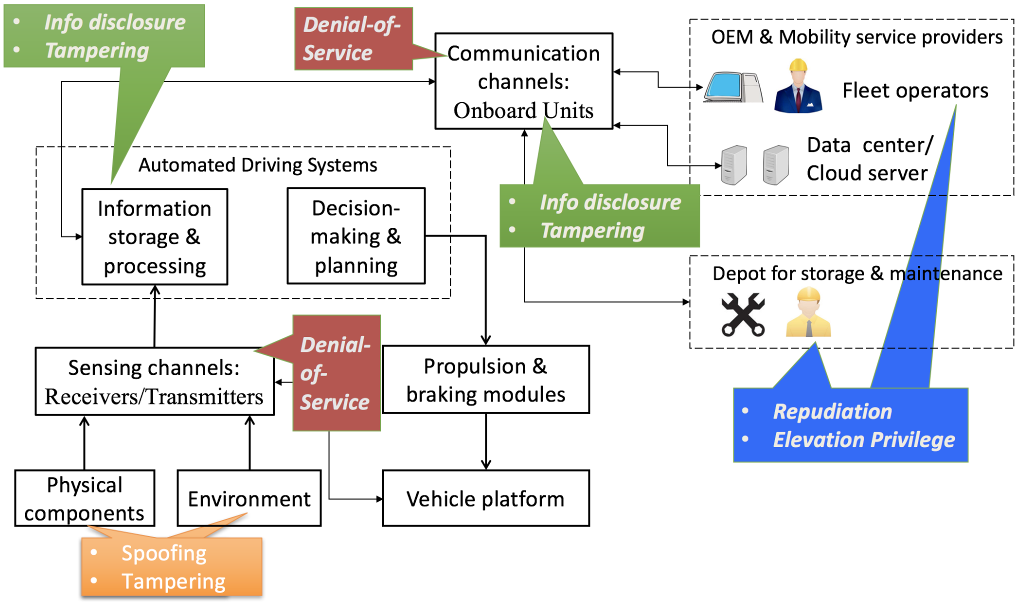
TMC.

3.3 Define attacker model and document assumptions

After identifying key components, principals, and stakeholders for the system architecture, the next task for security engineers and technical experts is to jointly derive attack models. Attack models refer to the assumptions on the power of an adversary (i.e., what an adversary can do), the knowledge and expertise (s)he has about the target system, the motivation for initiating attacks, and sometimes the characteristics of a certain social group (a crime unity) to which the adversary belongs [99]. For example, an adversary can be a car owner [1] who wants to gain economic benefits from selfish behaviors, a technician installing ransomware on the victim's vehicle, or terrorists trying to sabotage CAVs or the transportation system.

To build realistic attack models, we need to consider the methods adversaries use to launch attacks. A conceptual model we developed in our previous work for security analysis in product designs [132] provides guidance in this aspect, as shown in Fig. 3-3. The model is based on Microsoft's STRIDE model [78] which is used in threat modeling for information systems and adapted to CAV development by merging STRIDE with control-theoretic analyses used in security [153, 67] and human performance analyses [108]. Each type of malicious behavior in the STRIDE categories, which include Spoofing, Tampering, non-Repudiation, Information Disclosure, Denial-of-Service, and Elevation privilege, influences the electronic or physical components of the CAV architecture in Fig. 3-3. Engineers then determine if any of these malicious behaviors would compromise CAV functionalities, disrupt services, or cause damage to assets.

To illustrate the conceptual model, we present an example of an emerging type of threats (adversarial machine learning) in the context of CAVs. This type of threat exploits the weakness of machine learning algorithms (ML), especially deep neural



Info disclosure
Tampering
Denial-of-Service
Communication channels: Onboard Units
OEM & Mobility service providers
Fleet operators
Data center/ Cloud server
Automated Driving Systems
Information storage & processing
Decision-making & planning
Info disclosure
Tampering
Depot for storage & maintenance
Sensing channels: Receivers/Transmitters
Denial-of-Service
Propulsion & braking modules
Repudiation
Elevation Privilege
Physical components
Environment
Vehicle platform
Spoofing
Tampering

Figure 3-3: A conceptual model for deriving attack behaviors extended from [132]

networks (DNNs), for perception tasks such as object detection and recognition. For example, a physical object with a small perturbation in terms of geometry shape, color, or orientation, which is imperceptible to human eyes, can fool the DNNs into misclassifying objects. In addition to causing false alarms to drivers (HE-2) and unintended maneuvers (HE-3) for Type-1 vehicles, adversarial attacks on a Type-2 CAV's perception system built from ML can also result in its violation of traffic rules and regulations (HE-4). It is demonstrated in [77] that a CAV can experience unintended acceleration and thus overspeed even if an adversary slightly perturbs the speed limit sign to fool the perception system.

In order to derive a comprehensive list of attack scenarios for adversarial attacks that cause HE-1 and HE-2, engineers can examine the path of information flow shown in Fig. 3-3, including signal generation and transmission in the physical environment, signal reception by receivers, information processing for generating training data for AI, data sharing and storage in the cloud. For example, three possible ways of attacking ML models in vision-based tasks based on the conceptual model include:

- Spoofing and tampering physical or digital objects in the environment.

- Spoofing sensing modules such as camera and LIDAR.

- Data poisoning of training data for DNNs through unauthorized access to cloud servers storing these data.

An adversary can spoof objects or the physical environment by creating an illusion of physical objects by projecting images of pedestrians or stop signs with equipment placed on roadside or drones [83]. The adversary can also tamper physical objects by placing paper stickers on traffic signs to fool DNNs into confusing the difference between a stop sign and a speed limit sign, which is demonstrated in both laboratory settings [35] and in the real world [77]. With small physical perturbations on traffic signs, the adversary can maximize the likelihood of classifications errors and in some cases trick ADAS to accelerate to 80 miles per hour (mph) even if the vehicle actually meets a 30 mph sign. Similar attacks are found to be effective in fooling the ADAS to misidentify lanes and cause the CAV to drive into the reverse lane [137]. In addition to camera-based perception systems, CAVs relying on LIDAR are also vulnerable to adversarial attacks [15].

Another way for an adversary to create adversarial examples is to target sensing modules directly such as LIDAR. It is demonstrated in [15] that injecting malicious signals into the reflected light pulses can create adversarial 3d point clouds that cause classification errors and thus unintended emergency braking (HE-3).

Additionally, an adversary who gets unauthorized access to training data for DNNs can conduct data poisoning [126, 10, 57]. Injecting false training data into the training stage of ML algorithms makes classifiers make similar classification mistakes. Since vision data for training DNNs is collected by each individual CAV but aggregated in the cloud server maintained by OEMs or MSPs, an adversary who gets access privileges (the “E” category in STRIDE) to either in-vehicle storage [101] or on-line servers [58] can initiate such attacks.

For the Type-1 CAV, a malicious driver can conduct local data poisoning by adding noise patterns to sensor data before they are uploaded to the cloud server. For example, an “evil mechanic” [99] who is responsible for vehicle-fleet maintenance has access to the unprotected CAN bus and ECUs to install malware. When it comes to the Type-2 CAV, an adversary can conduct perturbations on the collective training dataset when (s)he has sufficient privileges to access on-line cloud servers. This concern arises when companies store their data on 3rd party cloud servers for computation-intensive tasks. For example, according to the report of the data breach affecting 57 million ride-hailing riders in 2016, the adversaries used stolen login credentials owned by valid employees to gain privileged access to data servers [84].

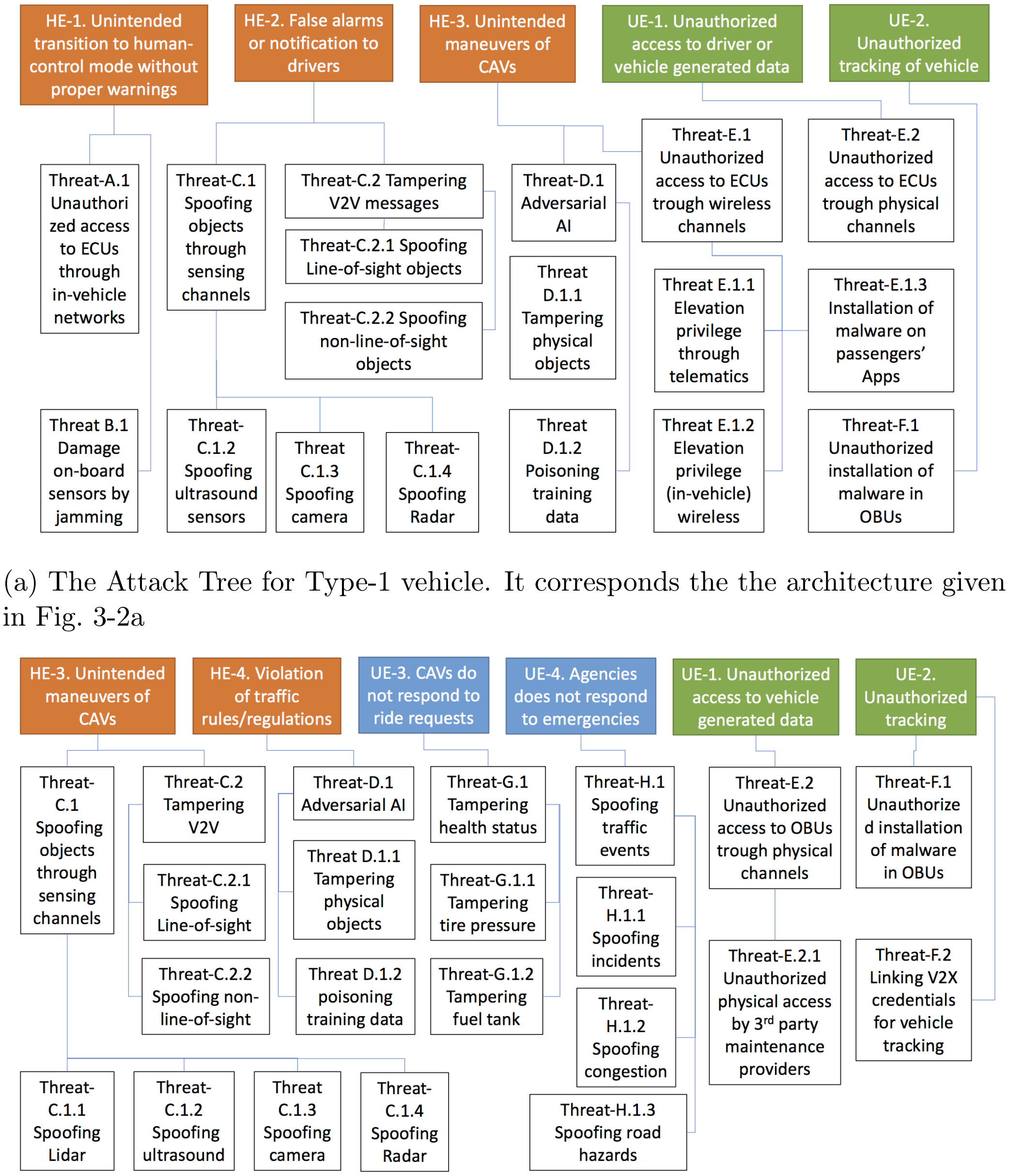
3.4 Derive Attack Trees for traceability

The final task is to document analysis results of identified threats and attack scenarios. The key is to establish traceability between high-level design goals and identified threats [132]. Here, we are interested in how a specific threat on on-board sensors, communication modules, or cloud servers can cause hazardous or undesired events previously defined.

We recommend Attack Trees [121] for this task. Threats are represented by leaves in Attack Trees while HEs and UEs are denoted by roots. This tree-based technique ensures that no high-risk threat will be omitted later in the implementation stage. The Attack Trees for organizing threats on Type-1 and Type-2 CAVs are given in Fig. 3-4.

Attack Trees introduce three benefits. First, the graphical forms help the three engineering teams manage design changes due to the adoption of new technologies. When an OEM uses new software and hardware modules to implement an old functionality [53], the Attack Trees visualize how these changes influence the results of the security analysis (i.e., traceability between HEs and threats). Second, the derived Attack Trees make it easier to prioritize among different design goals and thus determine which threat to deal with first. Third, the derived Attack Trees enable the estimation of the total cost of security defense mechanisms, i.e., the time and manpower allocated to dealing with identified threats.

We now give details of different types of threats based on the structure indicated by the Attack Tree in Fig. 3-4, which highlights the different attack surfaces between Type-1 and Type-2 CAVs.


HE-1. Unintended transition to human-control mode without proper warnings
HE-2. False alarms or notification to drivers
HE-3. Unintended maneuvers of CAVs
UE-1. Unauthorized access to driver or vehicle generated data
UE-2. Unauthorized tracking of vehicle
Threat-A.1 Unauthorized access to ECUs through in-vehicle networks
Threat-C.1 Spoofing objects through sensing channels
Threat-C.2 Tampering V2V messages
Threat-C.2.1 Spoofing Line-of-sight objects
Threat-C.2.2 Spoofing non-line-of-sight objects
Threat-D.1 Adversarial AI
Threat D.1.1 Tampering physical objects
Threat-E.1 Unauthorized access to ECUs trough wireless channels
Threat-E.2 Unauthorized access to ECUs trough physical channels
Threat E.1.1 Elevation privilege through telematics
Threat-E.1.3 Installation of malware on passengers' Apps
Threat B.1 Damage on-board sensors by jamming
Threat-C.1.2 Spoofing ultrasound sensors
Threat C.1.3 Spoofing camera
Threat-C.1.4 Spoofing Radar
Threat D.1.2 Poisoning training data
Threat E.1.2 Elevation privilege (in-vehicle) wireless
Threat-F.1 Unauthorized installation of malware in OBUs

(a) The Attack Tree for Type-1 vehicle. It corresponds the the architecture given in Fig. 3-2a

HE-3. Unintended maneuvers of CAVs
HE-4. Violation of traffic rules/regulations
UE-3. CAVs do not respond to ride requests
UE-4. Agencies does not respond to emergencies
UE-1. Unauthorized access to vehicle generated data
UE-2. Unauthorized tracking
Threat-C.1 Spoofing objects through sensing channels
Threat-C.2 Tampering V2V
Threat-C.2.1 Spoofing Line-of-sight
Threat-C.2.2 Spoofing non-line-of-sight
Threat-D.1 Adversarial AI
Threat D.1.1 Tampering physical objects
Threat D.1.2 poisoning training data
Threat-G.1 Tampering health status
Threat-G.1.1 Tampering tire pressure
Threat-G.1.2 Tampering fuel tank
Threat-H.1 Spoofing traffic events
Threat-H.1.1 Spoofing incidents
Threat-H.1.2 Spoofing congestion
Threat-H.1.3 Spoofing road hazards
Threat-E.2 Unauthorized access to OBUs trough physical channels
Threat-E.2.1 Unauthorized physical access by 3rd party maintenance providers
Threat-F.1 Unauthorized installation of malware in OBUs
Threat-F.2 Linking V2X credentials for vehicle tracking
Threat-C.1.1 Spoofing Lidar
Threat-C.1.2 Spoofing ultrasound
Threat C.1.3 Spoofing camera
Threat-C.1.4 Spoofing Radar

(b) The Attack Tree for Type-2 vehicle. It corresponds the the architecture given in Fig. 3-2b

Figure 3-4: The Attack Trees for Type-1 and 2 connected and automated vehicles.

3.4.1 Attack Trees for hazardous events regarding safety

To derive the Attack Trees for a HE, the security engineering team iterates through relevant system components to identify potential threats. System components can be found in CAV architectures described earlier, which include in-vehicle mechatronics platforms, on-board sensors and actuators, on-board communication modules, human-computer interface (for Type-1), computer controllers within vehicles or cloud servers, and physical or wireless links. For example, HE-3 of unintended maneuvers might occur during the braking, acceleration, or steering by the automated–driving system in Fig. 3-4b. Any unauthorized access and tampering with on-board sensing or V2V communication can cause this HE if the related information is used by the ADS for driving maneuvers.

To use the same conceptual model in Fig. 3-3) to derive detailed threats for the leaf nodes of the Attack Trees for HE-3, security engineers need input from CAV design teams and sensing module experts for the selection of sensing modules and communication modules used for ranging and object recognition. For example, range sensors that use electromagnetic or mechanical waves may be subject to signal relaying, jamming, or injection attacks [103, 123, 15]. Furthermore, perception systems based on visual clues (e.g., RGB information from photos or videos) from camera are vulnerable to adversarial attacks on the physical environment. Example Attack Tree for HE-3 and the list of physical threats on three types of on-board sensors, including LIDAR, ultrasound, and camera, are given in Fig. 3-4.

Cyber threats on LIDAR LIDAR, the acronym for Light Detection and Ranging, is a method to detect the relative distance between objects. Commercial LIDAR products that are found to be vulnerable to spoofing (inject fake signals in physical channels) and jamming (denial-of-service) attacks are mostly built on the time-of-flight (ToF) principle. A LIDAR mainly consists of a laser source, a transmitter, a receiver, and a signal processing module. The laser source generates short light pulses that are directed by the transmitter towards a given object (e.g., vehicles in front). Since a proportion of light pulses will be reflected back after they hit the object and are captured by the receiver, the signal processing unit determines the distance by multiplying the one-way time-of-flight with the speed of light [128]. Often, there will be a short time window (e.g., nanoseconds scale) for the receiver module to wait for the reflected light signal after the firing of light pulses. However, if an adversary can take advantage of this short-time window to inject signals before the reflected echoes, s(he) can fool the signal processing unit into thinking that the fake echoes are real ones reflected by the object. For example, Petit et al. conduct jamming and spoofing attacks on a commercial product Lux 3 manufactured by ibeo to introduce fake dots that are further away from the attacker's position [102]. Shin et al. take one step further by injecting 10 fake dots that are closer to the target than the attacker on a VLP-16 LIDAR manufactured by Velodyne [123]. Cao et al. fake 60 points on a VLP-16 by an enhanced attacking platform [7]. Additionally, the possibility of unintentional interference between two LIDAR sensors has been also evaluated by researchers, raising more concerns over LIDAR security vulnerabilities.

Cyber threats on Ultrasonic sensors An Ultrasonic sensor used in low-speed application scenarios, such as automatic parking systems, can become the target of jamming and spoofing attacks [150, 149]. Ultrasonic sensors are also built on the ToF principle, which counts the time it takes for ultrasonic pulses generated by the transmitter to travel back to the receiver. In addition to spoofing attacks on Ultrasonic sensors targeting the same vulnerability in the transmitter and signal processing module as in LIDAR, jamming attacks exploit the resonant frequency of the membrane in the receiver of an ultrasonic sensor. Ultrasound noise at the resonance frequency can create continuous vibrations in the membrane and disable the sensor.

Cyber threats on Camera Camera-based vision systems are used in object detection and recognition, tracking, and semantic segmentation tasks. This type of sensing solution is subject to denial-of-service (blinding) attacks. Just as regular cameras can be dazzled by the glare of the sun [9], some on-board camera in CAVs can be "blinded" by aiming a light source such as laser beam or LED (with either bursts of light or a constant beam) at the receiver lens [102]. In response, the digital camera will try to adjust its exposure automatically in order to adapt to the shift in the tonal range. Besides, strong light beams can cause permanent damage to the CMOS/CCD chip on the camera used in commercial CAVs [150].

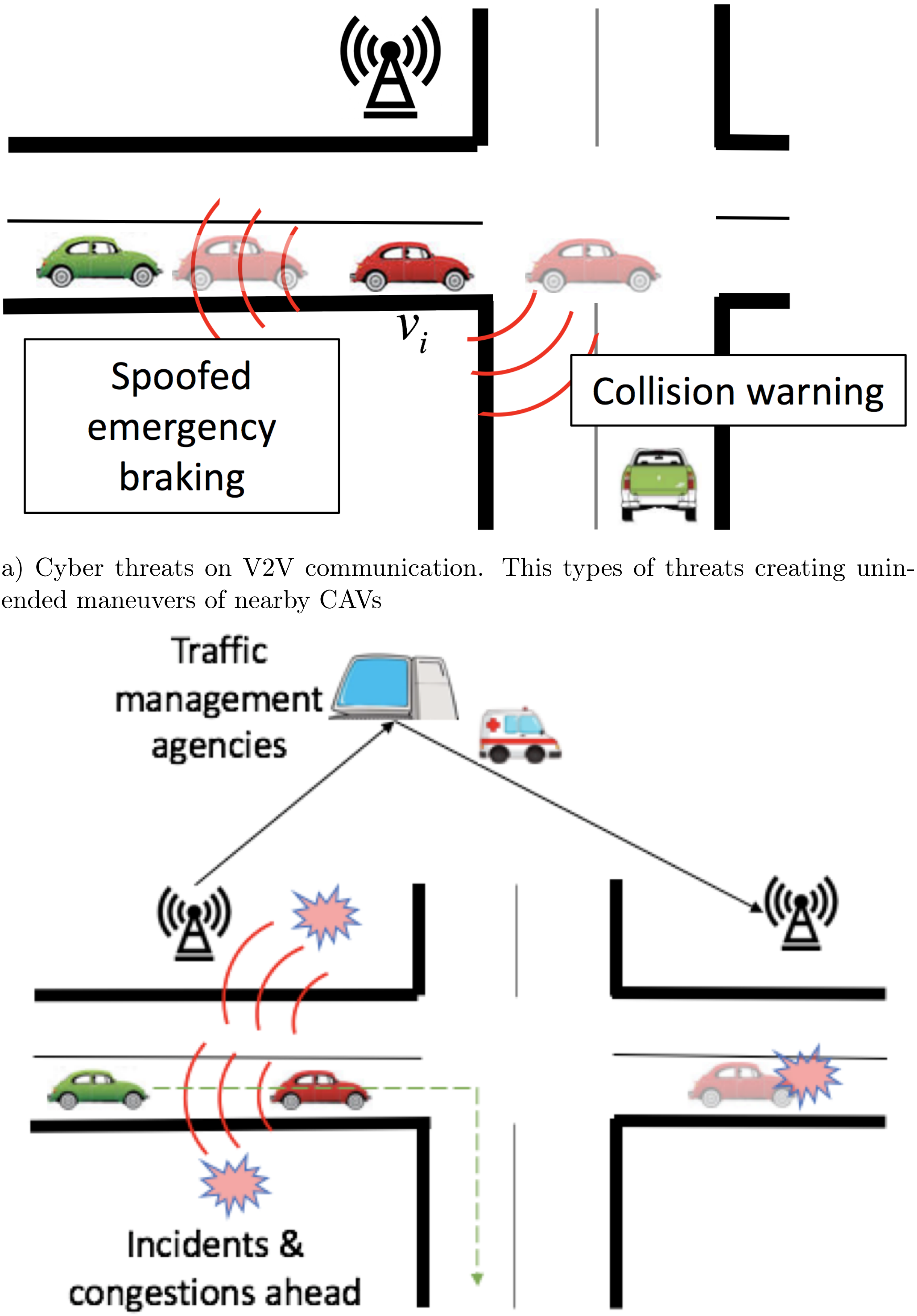
Cyber threats on V2V communication An adversary who holds valid V2X credentials or has control over a vehicle can send fake messages about vehicle locations and movements, which creates unnecessary warnings or emergency braking for nearby CAVs. These "Ghost" vehicles can be located at line-of-sight or non-line-of-sight areas with respect to the target vehicle of the attack. In the line-of-sight scenario, the maneuver of the target vehicle can be influenced by emergency braking signals [100], turn signals, or even emergency signals broadcasted by police cars. The non-line-of-sight scenario often occurs when an adversary spoofs ghost vehicles approaching a non-signal intersection. One example is the stationary attacker [120] who uses a wireless device with valid credentials [12] to broadcast fake messages to other vehicles indicating a collision risk, as shown in Fig. 3-5a. This threat can pose dangers to CAVs who emergency braking functions rely on V2V information.

This type of threat on V2V can be mitigated by either data plausibility checking (also named local misbehavior detection) or multi-modality data fusion techniques. The former category has been extensively studied in [11, 87, 7, 125]. Sensor fusion approaches are crucial for Type-2 CAVs, which are more self-sufficient in the sense that they do not rely on information from a single channel (e.g., V2V) for decision-making. We will elaborate this mitigation strategy in later sections.

3.4.2 Attack Trees for undesired events regarding serviceability

There are more attack potentials and security concerns over serviceability for Type-2 than Type-1 vehicles, because of the additional attack surfaces below.

An adversary can initiate attacks to compromise two types of services: ride-hailing and ride-sharing services by mobility-sharing companies (UE-3) and road emergency responses (UE-4) by emergency responders, such as law enforcement and medical agencies, as shown in Fig. 3-5b.


Spoofed emergency braking
v_i
Collision warning
Traffic management agencies
Incidents & congestions ahead

(a) Cyber threats on V2V communication. This types of threats creating unintended maneuvers of nearby CAVs

(b) Cyber threats on V2I communication. This type of threats create denial of service in private mobility services or public emergency responses

Figure 3-5: Graphical illustrations of cyber threats on V2X

Cyber threats on private mobility-sharing services Ride-sharing services supported by CAVs can become the target of denial-of-service attacks [25, 1, 110, 140]. An inside adversary can send false information about vehicle locations and health status to the cloud server to confuse the central dispatcher. Health information may include engine status, tire pressure (Threat-G.1.1) [50], battery status, fuel level (Threat-G.1.2), or about faults in ECUs. Since the dispatching algorithm needs information to determine which CAV in the fleet should be assigned to a given ride, false information can hinder fleet scheduling. For example, spoofed alert packets that are sent to the tire-pressure monitoring system can indirectly influence fleet management.

Cyber threats on public emergency responders For CAV-based transportation services, an adversary can fool the traffic control and management system by sending false reports about incidents [38] or congestion [156]. As a result, the automatic traffic controller may switch to wrong traffic signals [39, 41] or request police cars and ambulances to incorrect locations (Threat-H.1). The need for ensuring the trustworthiness of vehicle-reported information become more urgent as more connected-vehicle applications are deployed in the real world. One example is probe data enabled traffic monitoring (PDETM), an application that transmits real-time traffic data from vehicles to the traffic management system. PDETM is proposed as one of the potential applications in one of the connected vehicle pilot deployment programs by U.S. DOT [135]. The traffic controller may "use probe data information obtained from vehicles in the network to support traffic operations, including incident detection and the implementation of localized operational strategies".

3.4.3 Attack Trees for undesired events regarding privacy

Attack Trees that cause unauthorized access to customer PII data (UE-1) and tracking of vehicle movement (UE-2), as shown in Fig 3-4, threaten customer privacy. A Type-2 CAV has potential threats on privacy than a Type-1 CAV does, because more stakeholders have the privilege of accessing the data, such as technicians for maintenance or fleet operators as mentioned earlier. Besides, the adoption of the

V2X technology for Type-2 CAVs also gives adversaries the opportunity to track the path of a vehicle's movement through V2I messages (Threat-F.2).

Cyber threats on access to customer data Unauthorized access to in-vehicle networks, one of the risks responsible for the loss of vehicle or customer data, is discussed in [71] and evaluated through experiments [61, 23]. These experiments include security analyses for attackers who gain physical access through OBD-II ports or telematics systems. The latter is demonstrated in a real-world scenario where an attacker hacks into the ECU controlling the braking function through infotainment systems [80].

For a Type-1 CAV, although it is arguable that an attacker (a person different from the driver) can easily get physical access due to its private ownership [23], the chance that a driver unintentionally installs uncertified third-party applications increases as the automotive industry develops more general-purpose operating systems to support telematics applications [49, 34]. For example, Google has collaborated with Intel and car manufactures including Audi and Volvo to develop an Android OS for vehicle infotainment systems [48].

It is also possible that an uncertified application is malware that invades customer privacy by reading data from sensors or PII data stored in the on-chip memory. Woo et al. demonstrate a practical attack on CAN bus through an OBD diagnostic tool. The malware OBD diagnostic tool is installed on drivers' smartphones and paired with the vehicle through Bluetooth [148]. The malicious App can give the control of ECUs to remote attackers.

For a Type-2 CAV, there exist additional modes of unauthorized access, such as when vehicle-generated and customer PII data are transmitted and stored on a cloud server. An mobility service provider can use the stored customer data for user authentication or payment when external individuals access to customer data stored in the server owned by a third-party cloud service provider [60]. Engineers need to consider when and how to encrypt and decrypt these data during the transmission to protect the confidentiality of customer data in case of data breaches.

Cyber threats on tracking vehicle movement The tracking of vehicle locations and movement results from the installation of malware in OBUs (Threat-F.1) or linking vehicle credentials included in V2X messages with location data (Threat-F.2). The latter has been demonstrated in [103] when an inside attacker deploys tracking devices on compromised RSUs to monitor a vehicle's trajectory history. The process of designing defenses to this threat illustrates how to resolve conflicts between security and privacy goals, as will be discussed further when we present mitigation solutions in section IV-C.

3.5 Modeling Language and Tool Support

To maintain traceability from design goals and detailed threats to mitigation solutions throughout the security engineering process, the interdisciplinary engineering teams need tools to ensure that changes made by one team (e.g., CAV design team) can be reflected immediately and consistently in the view of other teams.

Tools developed from general modeling languages such as UML [43] or SysML [44] can help in this process. Software tools based on these formal languages help engineers visualize and manage the proposed security engineering process. Specifically, Attack Trees documenting analysis results can be serialized and stored in enterprise cloud servers. Maintaining a "single" model throughout CAV life-cycle enables multiple teams to have a consistent view of the gap between unresolved threats and existing mitigation solutions implemented.

The key task for developing tools to support the security engineering process is to define mathematically the core concepts of threat modeling discussed earlier such that they can be transformed into graphical models in the SysML language. Two examples of HEs and threats are given below.

- *Definition. 1.* A HE is a four-tuple $< F, F^T, Co, AI >$ where F is a function that CAV supports, F^T denotes the failure modes for F, Co represents the environment conditions under which the hazardous events occur, AI is a three-tuple $< Se, C, E >$ which represent the severity, controllability, and exposure

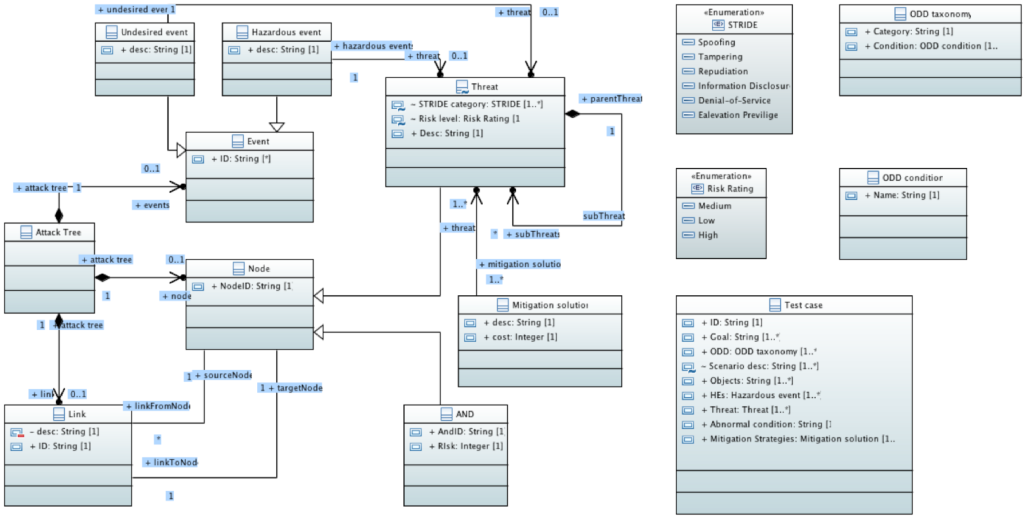
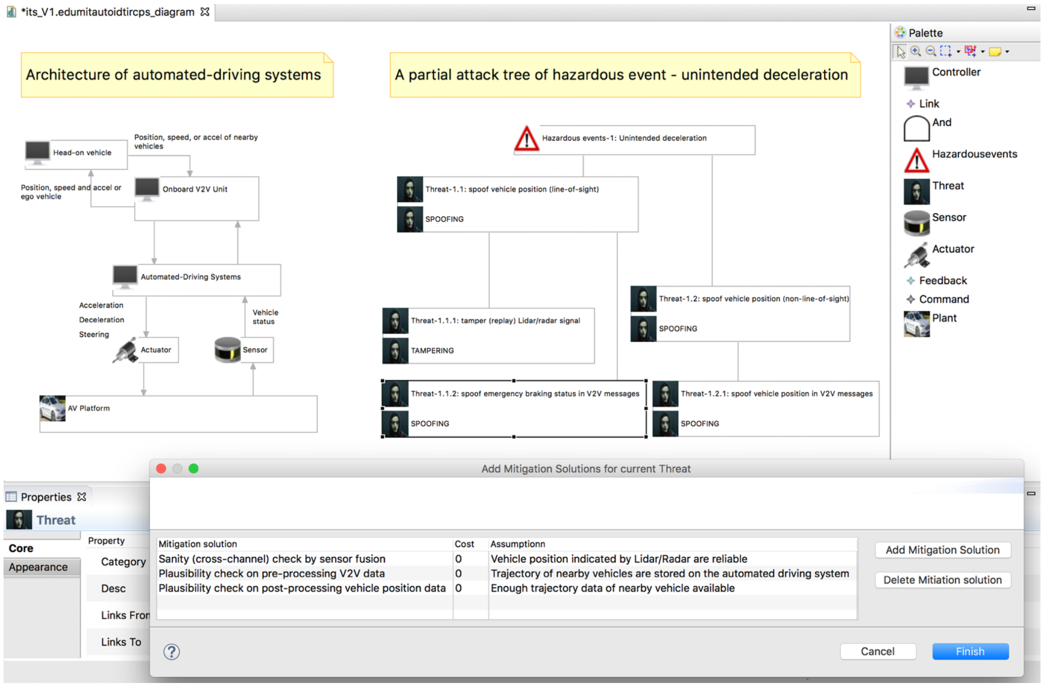


Figure 3-6: SysML model for realizing the proposed security engineering process in Fig. 3-1

of the hazardous event defined.

- *Definition. 2.* A Threat (T) is a five tuple $< Tc, Fb, Fa, P, S >$ where Tc is the threat category in the STRIDE model, Fb represents information from sensors or V2X modules that a given threat is related, Fa is the parent node that can be a HE or a Threat, P is the probability that this particular threat can occur, which needs to be judged by domain experts, S is a child node of Threats.

After formally defining security, privacy, and privacy concepts in the proposed engineering process, engineers can develop corresponding SysML models and toolsets. Fig. 3-6 and 3-7 show the SysML language model and visualization tool that we developed to realize the security engineering process proposed in this chapter. It helps in maintaining a single model of security and safety analyses shared by different engineering teams can avoid inconsistencies between requirements and designs during CAV development.


*its_V1.edumitautoidtircps_diagram
Architecture of automated-driving systems
A partial attack tree of hazardous event - unintended deceleration
Head-on vehicle
Position, speed, or accel of nearby vehicles
Position, speed and accel or ego vehicle
Onboard V2V Unit
Automated-Driving Systems
Acceleration
Deceleration
Steering
Vehicle status
Actuator
Sensor
AV Platform
Hazardous events-1: Unintended deceleration
Threat-1.1: spoof vehicle position (line-of-sight)
SPOOFING
Threat-1.1.1: tamper (replay) Lidar/radar signal
TAMPERING
Threat-1.1.2: spoof emergency braking status in V2V messages
SPOOFING
Threat-1.2: spoof vehicle position (non-line-of-sight)
SPOOFING
Threat-1.2.1: spoof vehicle position in V2V messages
SPOOFING
Palette
Controller
Link
And
Hazardousevents
Threat
Sensor
Actuator
Feedback
Command
Plant
Properties
Threat
Core
Appearance
Property
Category
Desc
Links Fron
Links To
Add Mitigation Solutions for current Threat
Add Mitigation Solution
Delete Mitiation solution
Cancel
Finish

Mitigation solution	Cost	Assumptionn
Sanity (cross-channel) check by sensor fusion	0	Vehicle position indicated by Lidar/Radar are reliable
Plausibility check on pre-processing V2V data	0	Trajectory of nearby vehicles are stored on the automated driving system
Plausibility check on post-processing vehicle position data	0	Enough trajectory data of nearby vehicle available

Figure 3-7: The visualization tool developed based on the SysML model presented

Chapter 4

Polarization-based object recognition for Mitigating Physical Threats to on-board sensing

The ability to sense and perceive nearby objects, such as pedestrians, bicyclists, and traffic signage, is crucial for safe decision-making by mobile robots. However, the use of commercial vision sensors such as camera and light detection and ranging (Lidar) in perception tasks suffers from the potential of misclassifying objects in extreme illumination (e.g., darkness, strong sunlight) or adverse weather conditions.

The death of a bicyclist during Uber's testing of its autonomous vehicles has made the company halt its efforts of testing activities, incurring huge financial losses to the company [85]. It was confirmed in the preliminary investigation report [85] by National Transportation Safety Board (NTSB) that the self-driving system had not issued emergency braking until 1.3 seconds before the impact even though an "unknown" object was registered on Lidar and radar 6 seconds before the collision. Although this incident can be attributed to design flaws in software algorithms according to NTSB, this incident indicates the urgent need to acquire more reliable information about the type and material property of nearby objects for safety-critical decisions by automated-driving systems.

Another incident that raises safety concerns over camera-based sensing solutions is the fatal crash of a semi-automated vehicle made by Tesla into a truck in 2019 Florida [86]. According to a statement by Tesla after the crash, "neither Autopilot nor the driver noticed the white side of the tractor trailer against a brightly lit sky, so the brake was not applied" [136]. Although it is presumptuous to draw conclusions as to the failure of the on-board camera used in the target vehicle model, the automated-control system can utilize additional data modalities providing more information about the surface property of nearby objects for safe driving maneuvers.

4.1 Depth, RGB and, polarization vision for object recognition

Although Lidar, which is built on the time-of-flight principle, can provide more precise information about nearby objects, it may only generate sparse 3D point of clouds for object recognition tasks when the distance between the object in-front and the ego-vehicle becomes large. Even worse, A Lidar can produce falsified signals in adverse environmental conditions (e.g., heavy rains, fog, or dust), leading to risky vehicle maneuvers.

We present a solution for improving the richness of information provided by optical sensors for safe-critical decision-making in autonomous-driving applications [130]. This method measures changes in polarization states between incident and reflected light from target objects so that their material and surface properties can be deduced. The automated-driving system can then merge polarization information with RGB and depth information to make more informed decisions regarding how to maneuver the car.

Although polarization techniques have been explored by researchers for classifying object and material (e.g., cars and trucks) [115, 47, 107], differentiating man-made and natural objects [146], and enhancing contrast to spot camouflaged objects [112], their applications to object recognition in safety-critical scenarios remains unsolved.

4.1.1 Light polarization

Light is an electromagnetic wave and can be described by the vibration of its electric field whose direction is perpendicular to the direction of light propagation [46]. The electric field can be further decomposed and described by two orthogonal vibrations, which is described as

$$\overrightarrow{E}_x(z,t) = \hat{i}E_{0x}cos(kz - \omega t) \tag{4.1}$$

$$\overrightarrow{E}_y(z,t) = \hat{i}E_{0y}cos(kz - \omega t + \varepsilon) \tag{4.2}$$

It is the ε, the relative phase difference between these two waves in 4.1 and 4.2, and E_{0x} and E_{0y}, the amplitude of electric fields, that determine the polarization states of light.

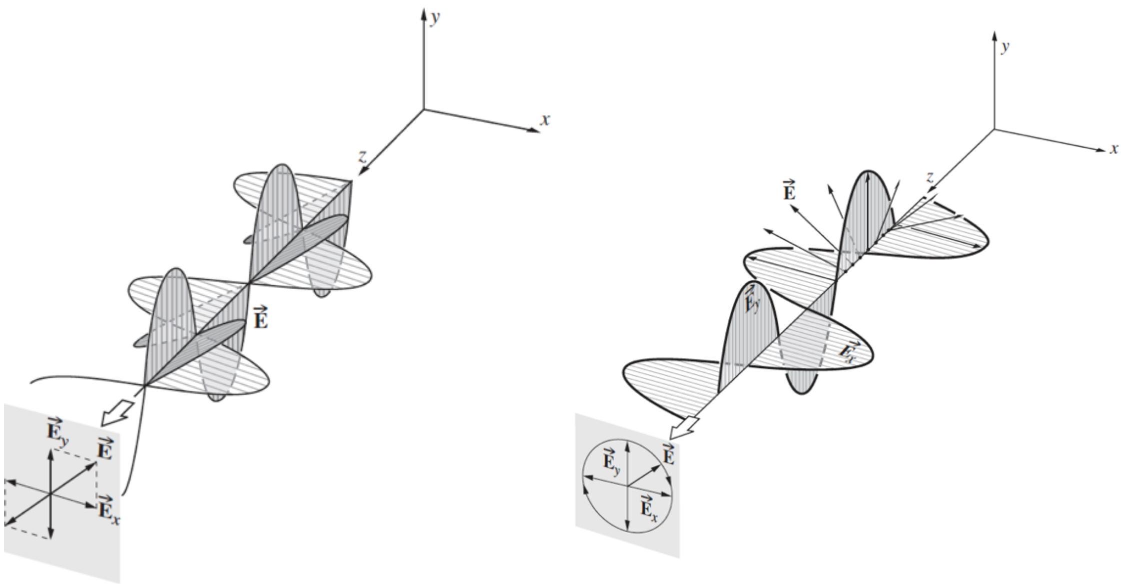
There are three categories of polarization states including linear, circular, and elliptical polarization. Specifically, a light wave is defined as linear polarized if $\varepsilon = 2\pi k$ where $k = 0, \pm1, \pm2, ...$, circular polarized if $E_{0x} = E_{0y}$ and $\varepsilon = \frac{-\pi}{2} + 2k\pi$ where $k = 0, \pm1, \pm2, ...$, or otherwise elliptical polarized.

4.1.2 Stokes treatment of polarization

The method of determining light polarization involves amplitude and phase between two orthogonal electrical fields $\overrightarrow{E}_x$ and $\overrightarrow{E}_y$, which are difficult to measure. From an application point of view, it will be beneficial to use something observable and quantifiable to represent the polarization state of light.

Stoke vectors were proposed by Stokes [127] and are functions of measurable parameters (i.e., intensity) of light. There are four components in Stoke vectors (equations 4.3) where E_{0x} and E_{0y} represent maximum amplitudes of the electrical field in the x and y directions.

$$S_0 = E_{0x}^2 + E_{0y}^2 \tag{4.3a}$$



(a) Linear polarization

(b) Circular polarization

Figure 4-1: Different polarization states of light [46]

$$S_1 = E_{0x}^2 - E_{0y}^2 \tag{4.3b}$$

$$S_2 = 2E_{0x}E_{0y}cos\varepsilon \tag{4.3c}$$

$$S_3 = 2E_{0x}E_{0y}sin\varepsilon \tag{4.3d}$$

S_0 denotes the total intensity of the optical wave, S_1 is the difference in intensity between the horizontal and vertical field of the optical wave, S_2 represents the intensity of linearly polarized light in the angles of 45 and -45, and S_3 represents the intensity of circularly polarized light.

Different methods for measuring stokes vectors have been developed. The "fixed quarter-wave" method [119] is used in this thesis to make the proof-of-concept experimental platform easy to set up in a benchtop. This method passes light through a retarder (often a quarter-wave plate) and a linear polarizer sequentially and measures the intensity of the light while rotating these two instruments to certain angles.

- Retarder: a retarder is an optical instrument that shift the phase difference between the $\overrightarrow{E}_x$ and $\overrightarrow{E}_y$ fields by a fixed amount (e.g., $\pm\frac{\pi}{4}$), as shown in Fig. 4-2. It has a two axes that are perpendicular to each other: A fast axis (x-axis

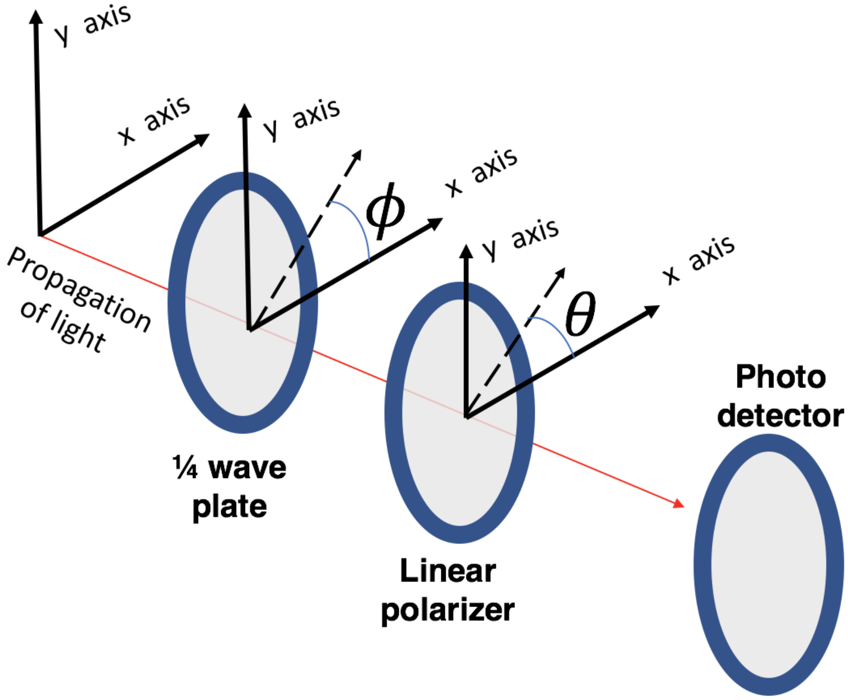


Figure 4-2: Fixed quarter-wave plate methods for measuring Stokes Vectors [119]

in the figure) that advances the phase of the field in that direction and a slow axis (y-axis in the figure) that maintains the phase of the field.

- Polarizer: a polarizer is a optical instrument that takes unpolarized light as input and outputs polarized light. A linear polarizer will have an "easy" axis that only allows optical waves with vibration aligned with the easy axis to pass through, as shown in Fig. 4-2.

The theoretical foundation for deriving Stokes vectors by measuring light intensity is governed by equation 4.4, where θ denotes the angle of the linear polarizer to the horizontal axis and ϕ denotes the angle of quarter wave plate to the horizontal axis [119].

$$I(\theta, \phi) = \frac{1}{2}(1, cos2\theta, sin2\theta cos\phi, -sin2\theta sin\phi)(S_0, S_1, S_2, S_3)^T \tag{4.4}$$

Based on 4.4, the relation between stoke vector and light intensity can be represented [119] as

$$S_0 = I(0^\circ, 0^\circ) + I(90^\circ, 0^\circ) \tag{4.5a}$$

$$S_1 = I(0^\circ, 0^\circ) - I(90^\circ, 0^\circ) \tag{4.5b}$$

$$S_2 = 2I(45^\circ, 0^\circ) - S_0 \tag{4.5c}$$

$$S_3 = S_0 - 2I(45^\circ, 90^\circ) \tag{4.5d}$$

For I(θ,ϕ), the intensity of the light wave of interest is measured by a photo detector after it passes through a quarter wave plate with the angle of its fast axis being equal to θ and linear polarizer with the angle of its easy axis being equal to ϕ, as shown in Fig. 4-2

4.2 Passive and active polarimetry

Polarimetry refers to the measurement of the polarization state of light and can be classified into two categories: Passive polarimetry which uses natural light (unpolarized) such as sunlight as the light source and measures the change in the polarization state of the reflected light from the target of interest and active polarimetry which generates polarized light by using a laser as the light source. Using a laser as the light source provides greater information richness about polarization properties of the target object [97].

The proposed approach developed in this thesis, multi-"polar" active polarimetry, extends previous works in active polarimetry [97] but adopts multi-wavelength laser sources each of which are modulated to generate multiple polarization states for target discrimination. Table 4.1 compares the proposed approach to previous passive and active approaches.

Table 4.1: Methods for object recognition based on the polarization property of light

Difference	**Passive polarimetry**	**Active polarimetry**	**Proposed approach**
Light source	Natural light	Single wavelength laser	Muli-wavelength lasers
Control over polarization of light source?	No	Yes	Yes, generate multiple polarization states for each wavelength
Mathematical principle	Fresnel reflection model	Stokes Vectors	Stokes Vectors
Application	Polarization camera	Lidar	Lidar
Literature	Wolf [146], Liu [72], Fan [37] Blin [14]	Chun [26], Pasqual [97] Queau [107]	Suo and Sarma [130]

4.3 Multi-"Polar" source active polarimetry

4.3.1 Design goals and the system concept

As shown in Fig. 4-3, the multi-"polar" polarimetry system includes a modulator for generating light sources with pre-determined polarization, an analyzer that measures the intensity of light based on equation 4.5, and a software module that calculates values of polarization variables, such as angle of polarization (AOP), degree of linear polarization (DOLP), and degree of polarization (DOP), and conducts polarization pattern matching based on changes in these variables.

As discussed earlier, the proposed approach involves the use of multiple light sources with different polarization states to "illuminate" the target. In the proof-of-concept design used for evaluation, three laser sources with x-polarized (horizontally), y-polarized (vertically), and circularly polarized are needed. Therefore, a polarization modulator is needed for controlling polarization-state generation. Its internal structure is shown on the bottom-left of Fig. 4-3. An unpolarized laser source is first linearly polarized by passing through a linear polarizer with a horizontal (x) axis. The generated polarized light is then split into three beams. The first beam is di-

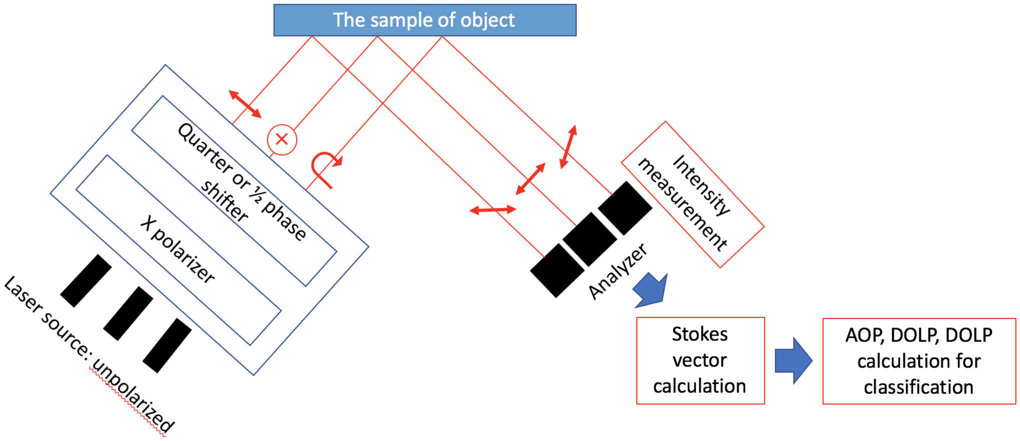

The sample of object
Quarter or ½ phase shifter
X polarizer
Laser source: unpolarized
Intensity measurement
Analyzer
Stokes vector calculation
AOP, DOLP, DOLP calculation for classification

Figure 4-3: A conceptual design of the multi-"polar" polarimetry presented in this thesis

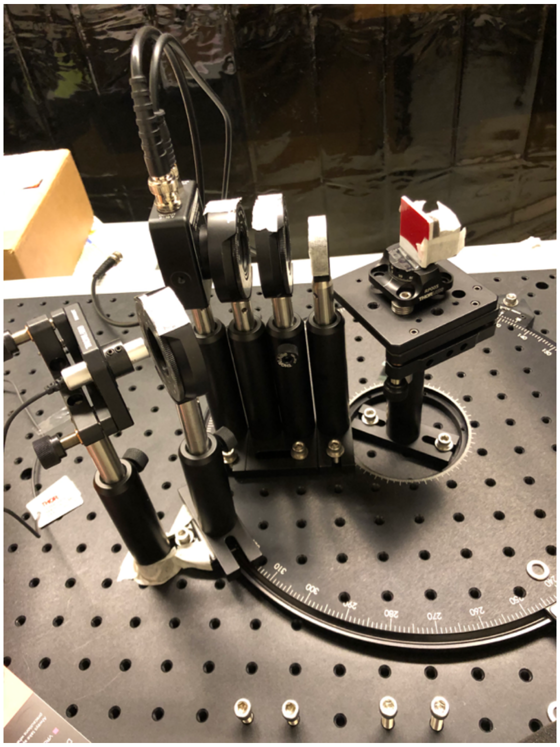


Figure 4-4: The experimental platform in a laboratory benchtop, based on the design in Fig. 4-3

rectly transmitted as the incident light. The second and third are passed through a half and quarter retarder (i.e., wave plate) respectively to generate y-polarised and circular polarized waves as light sources.

4.3.2 Experimental evaluation

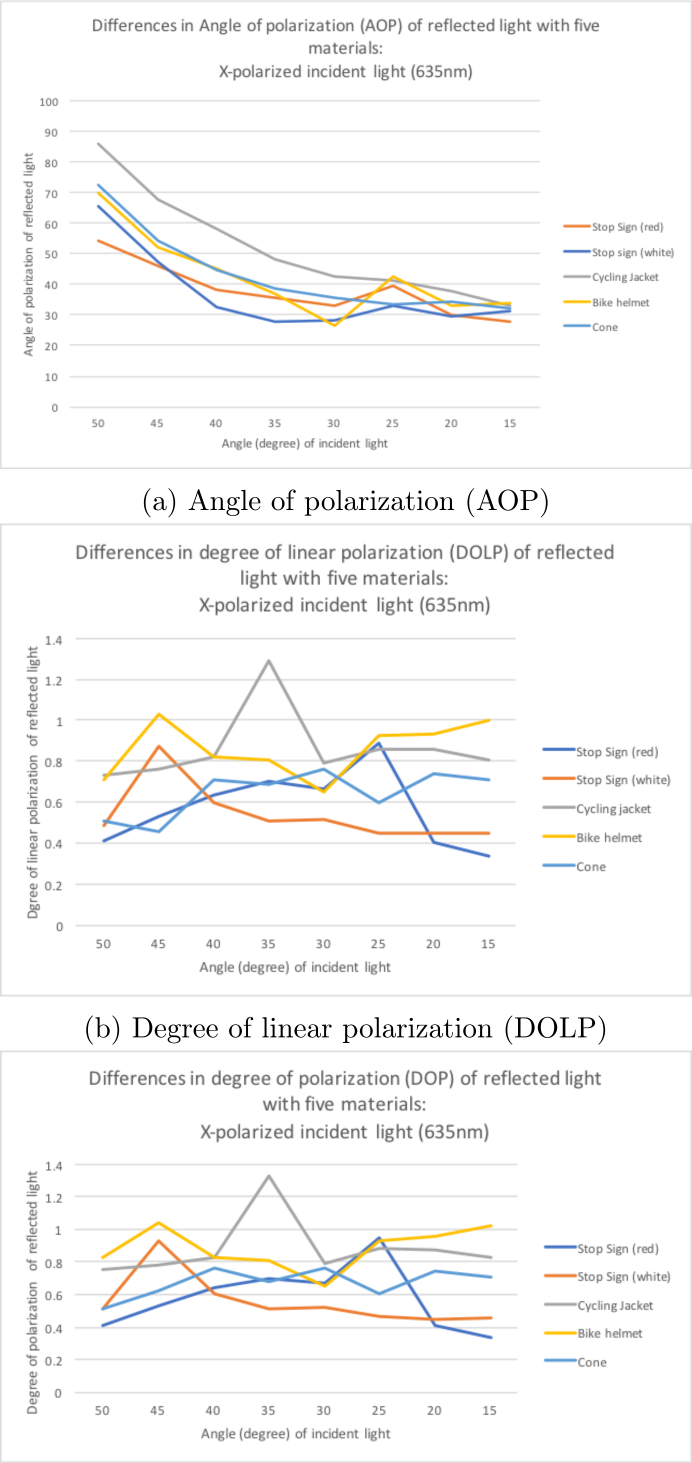
To evaluate the proposed approach, a proof-of-concept platform was built on a laboratory benchtop, as shown in Fig. 4-4. Five objects that often appear in the driving environment are selected for testing, including

- Stop sign (red region)
- Stop sign (white region)
- Cone (as a traffic sign)
- Cycling jacket wear by bycyclist
- Bike helmet

The goal is to compare the effect on polarization states of laser sources by different object surfaces. To take into account the variations in the incident angle for real-world applications, incident angles in the range between 15 and 50 degrees are considered. The results are shown in Fig. 4-5,4-6, and 4-7.

Insights from the experimental evaluation can help engineers design more robust optical sensing systems by using optical polarization techniques: The discriminability for testing samples by using different polarized laser sources and the polarization properties of different object surfaces.

- Discriminability for all object surfaces: When the laser source is X-linearly polarized, the system can achieve better discriminability regarding the difference in DOLP and DOP of reflected light when the incident angle is less than 20 degrees. On the other hand, the system that uses circular polarized or y-polarized light doesn't achieve as much discriminability for all object surfaces as x-polarized light.
- Discriminability for a single object surface: When the laser source is y-polarized laser source, the polarization state of reflected light from the stop sign (white region) in terms of DOLP and DOP is significantly different from other object

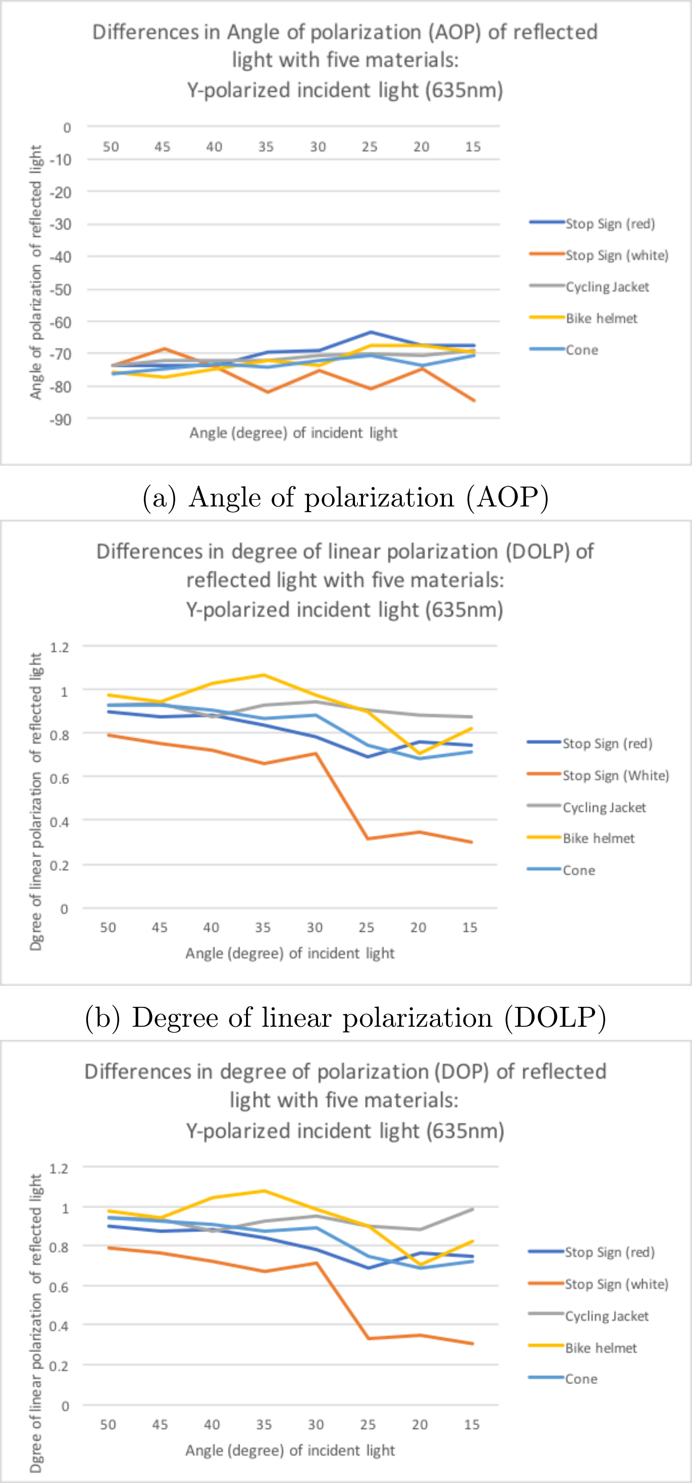


(a) Angle of polarization (AOP)

(b) Degree of linear polarization (DOLP)

(c) Degree of polarization (DOP)

Figure 4-5: Difference in discriminability for reflected light when laser source is x-linear polarized



(a) Angle of polarization (AOP)

(b) Degree of linear polarization (DOLP)

(c) Degree of polarization (DOP)

Figure 4-6: Difference in discriminability for reflected light when laser source is y-linear polarized

surfaces. Specifically, the stop sign (white region) exhibits a much greater depolarization effect than other objects with incident angles less than 30 degrees.

- Useful polarization property of particular materials. Polycarbonate (bike helmet) maintains DOLP and DOP while rubber (traffic cone) dramatically reduces them. In particular, polycarbonate maintains DOP for all sources of polarized light, making it an ideal candidate for making protective wearables, which are easily discerned by an optical instrument that can measure the exact quantity of degree of polarization.

4.4 Optimize polarization sensor designs

The solution developed in this chapter for mitigating environmental hazards such as darkness is promising in that the polarization information about object surface or material properties can improve the accuracy of current methods used in object recognition tasks such as deep neural networks. However, engineering trade-offs must be made between the desired discriminability among objects of interests and the cost of building optical polarization sensors. In particular, we suggest that designing polarization-based sensors for object recognition corresponds to solving the optimization problems.

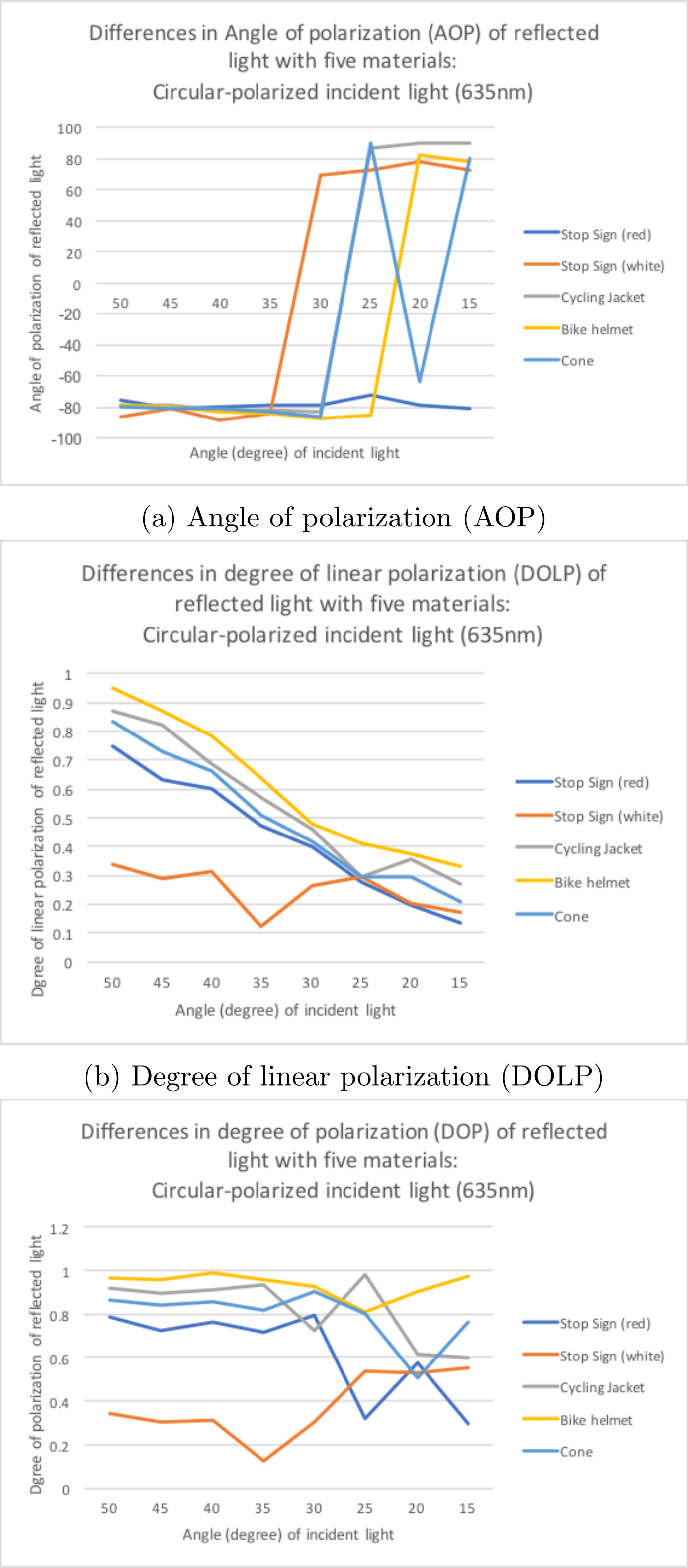
Decision variables There are three types of decisions that engineers must make. First, they must decide whether to adopt a laser source in a particular wavelength, denoted as ls_w. Second, the decision must be made on whether to adopt a set of linear polarizers that are sensitive to a given wavelength (e.g., w), which is denoted as lp_i^w. Third, similar to polarizers, engineers must decide whether to adopt a set of phase retarders that are sensitive to a given wavelength, which is denoted as rt_i^w. It should be noted that one particular polarizer or retarder may work for multiple laser sources in different wavelengths. However, this condition won't change our problem formulation.

Objective function Minimizing the total cost of optical polarization sensors. The total cost consists of fixed cost and variable cost. The fixed cost mainly involves the mechanical design that can hold all the lenses and laser sources. The variable cost consists of three parts, including the cost of lasers sources, the cost of linear polarizers and retarders. The object function that includes only the variable cost is given in 4.6.

$$\sum_{W1}^{Wn} ls_w + \sum_{i,w} lp_i^w + \sum_{i,w} rt_i^w \tag{4.6}$$

Constraints The only constraint can be expressed as "whether a particular pola-modality is needed" to differentiate between two object surfaces. If we use m_j to represent this constraint, then $m_j = 0$ means pola-state j is needed to differentiate between two object surfaces, and vice versa. Therefore, we have the equation 4.7, which means if a particular pola state is required, then engineers must be laser source, along with polarizers and retarders for generating that polarization state.

$$ls_w + lp_i^w + rt_i^w >= m_j \tag{4.7}$$



(a) Angle of polarization (AOP)

(b) Degree of linear polarization (DOLP)

(c) Degree of polarization (DOP)

Figure 4-7: Difference in discriminability for reflected light when laser source is circular polarized

Chapter 5

A Proof-of-Travel Protocol for Mitigating Cyber Threats on V2X Communication

While the mitigation solution presented in chapter 4 can improve the robustness of on-board sensing and mitigate physical threats due to extreme environmental conditions, it cannot deal with cyber threats involving tampering with and spoofing V2X messages. This chapter presents two methods to determine the trustworthiness of V2X messages: Vehicle-based trust management (for V2V communication) and infrastructure-based trust management (for V2I communication).

The former approach focuses on how to use vehicle reports about malicious V2V messages to determine the trustworthiness of a vehicle in a local region. It can be used for V2X credential management but is unable to identify and mitigate inside adversaries who use valid vehicle credentials to forge and disseminates traffic events through V2I channels. The second approach, a Proof-of-Travel protocol, builds on the support from infrastructure components (e.g., RSUs). Specifically, it rejects or accepts a V2I traffic event based on the reputation of the message origin, which is determined by the vehicle's travel behaviors observed by RSUs. Before presenting details of the protocols, we briefly review the security credential management in V2X communication.

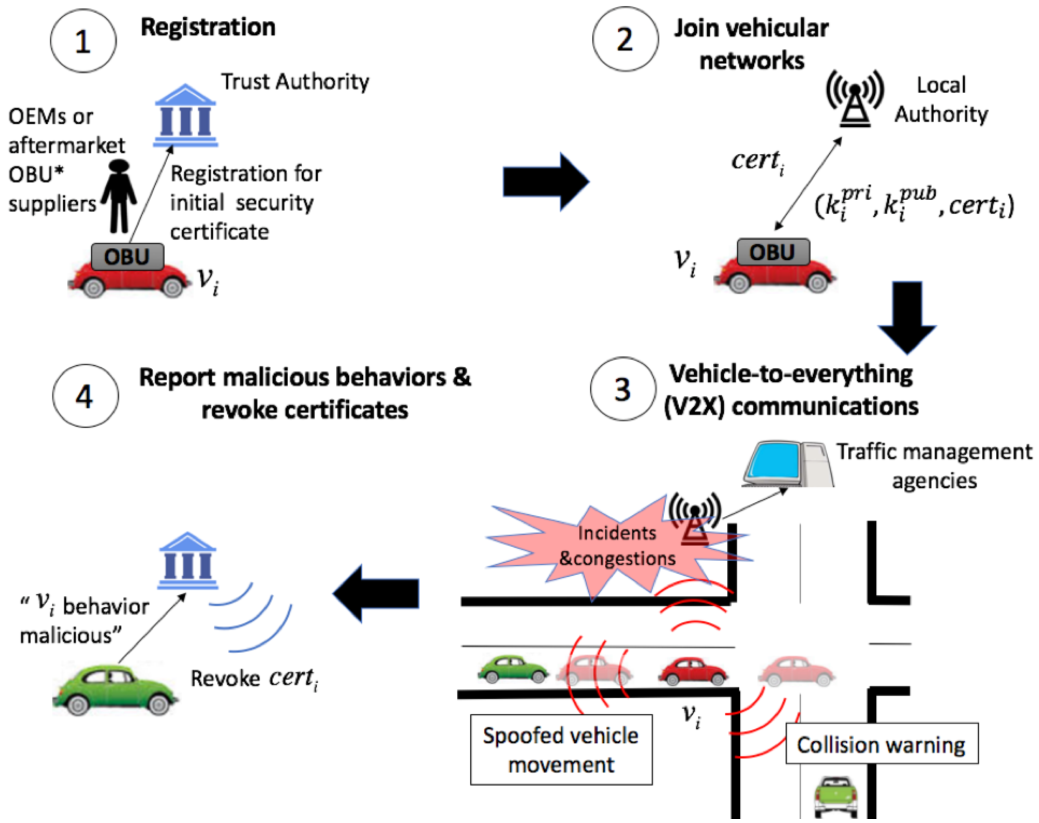

1
Registration
Trust Authority
OEMs or aftermarket OBU* suppliers
Registration for initial security certificate
OBU
v_i
2
Join vehicular networks
Local Authority
$cert_i$
$(k_i^{pri}, k_i^{pub}, cert_i)$
v_i
OBU
3
Vehicle-to-everything (V2X) communications
Traffic management agencies
Incidents &congestions
Spoofed vehicle movement
v_i
Collision warning
4
Report malicious behaviors & revoke certificates
" v_i behavior malicious"
Revoke $cert_i$

Figure 5-1: A simplified life-cycle of connected and automated vehicles [129]

5.1 An introduction to credential management in V2X communication

We assume that public-key infrastructure [158] is adopted to support vehicle credential management in vehicular networks. A trust authority (TA), set up and maintained by local or regional transportation agencies, is responsible for issuing and revoking vehicle credentials for V2X communication. Fig. 5-1 gives an overview of the four stages of V2X communication.

The example presented here is just for illustrating potential security vulnerabilities of V2X communication and may not perfectly replicate V2X credential management in real-world applications. We make reasonable assumptions based on previous publications and on-going test activities regarding connected vehicles in the U.S [88].

- Registering vehicles with the TA. The on-board unit (OBU) that restore certificates or credentials for V2X communications will be pre-loaded with security management certificates [158] as initial credentials upon registration. Note we use certificates to refer to the long-term credential materials for V2X authentication in this chapter. This can be accomplished by OEMs for future connected vehicles. Currently in the U.S., for the connected vehicle pilot projects conducted by the department of transportation (DOT) [88], OBUs made by automotive suppliers are pre-loaded with credentials and will be installed in vehicles after drivers finish registration with the local transportation department.

- Joining the vehicular network. When a vehicle v_i wants to join a local or regional vehicular network, it will use the long-term certificate to acquire a short-term credential, i.e., cryptography key pairs (a public key pk_i^{pub}, pk_i^{pri}) from a local TA such as a roadside unit (RSU). It is worth mentioning that giving infrastructure components the ability to track vehicle behaviors within networks can result in privacy risks where an adversary can use compromised RSUs to track vehicle trajectory for re-identifying passengers and drivers [103]. However, this cyber risk won't be the focus of this chapter.

- Communicating through V2X. During vehicle operation, a connected vehicle will use the key-pairs and certificates for encryption, decryption, signing and verifying V2X messages from other vehicles and infrastructure components. A key pair for vehicle v_i is asymmetric in the sense that v_i will use its private key pk_i^{pri} to sign V2X messages m_i to generate a signature $\sigma(m_i)$ before broadcasting it to nearby vehicles and RSUs. Other vehicles, without knowing the private key that v_i holds, can only use v_i's public key pk_i^{pub} for authentication.

- Reporting malicious behaviors for revoking certificates. Each connected vehicle in the network can report behaviors that are suspected to be malicious to the TA who will then determine whether to revoke certificates or not based on these reports. In vehicle-based trust management, reports include the evaluation of how trustworthy the target vehicle v_i is[55]. The TA may revoke certificates assigned to the target vehicle if the aggregated trust value on the target one is below a pre-determined threshold. On the other hand, evaluation information of v_i's trustworthiness is from RSRs in the second method of trust management.

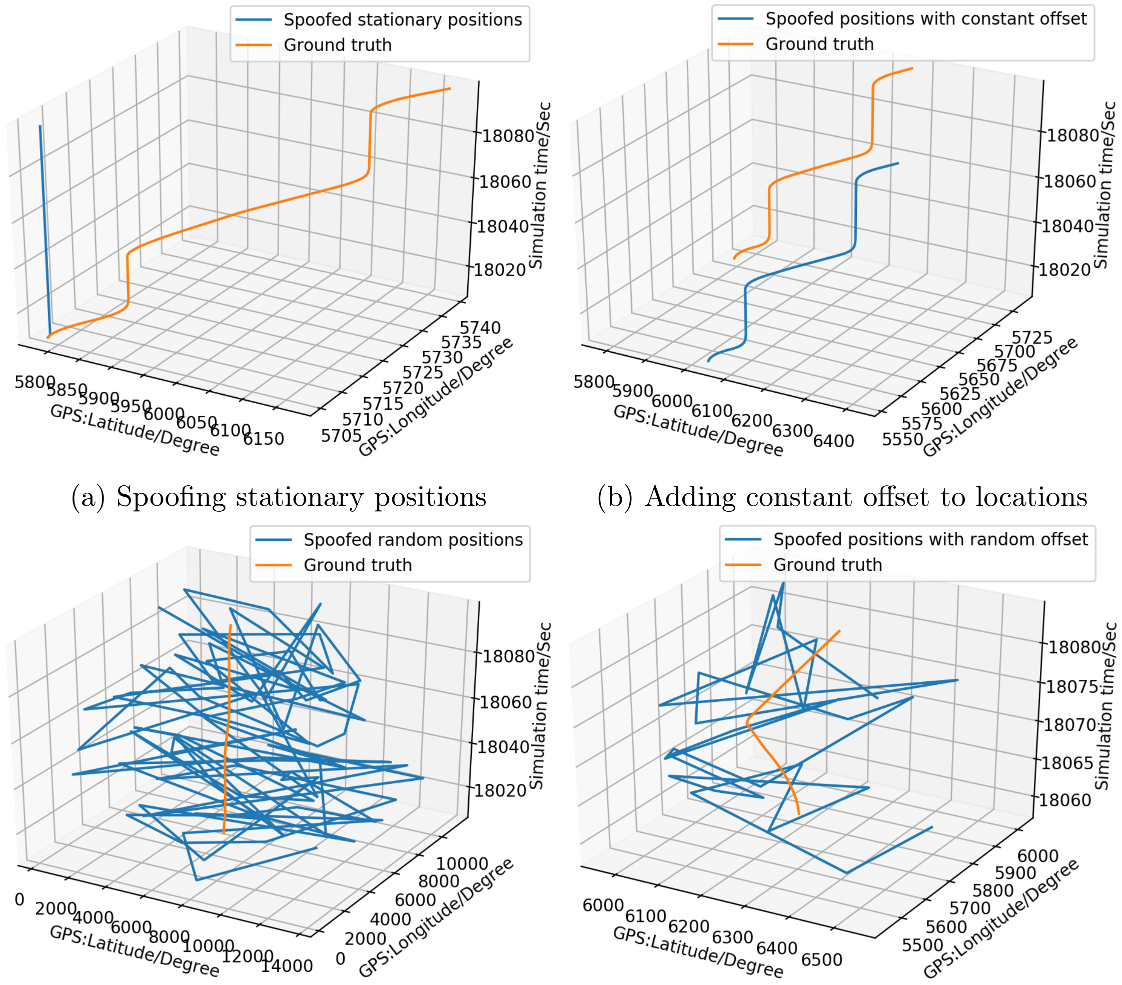
5.2 Problem formulation

There are two related security objectives that mechanisms presented in this chapter target: first, how does a TA quantifies and determines the trustworthiness of a given vehicle based on its message exchanges with nearby vehicles and infrastructure components through vehicle-to-everything (V2X) communication? Second, how does a TA manages V2X credentials assigned to a given vehicle based on its perceived trustworthiness.

To better understand the communication process, it is necessary to mention the assumptions the thesis makes on the format of V2X messages. There are industry standards and practices regulating the message exchange between vehicles and external entities. For example, SAE J2735 [56] defines the content and format of Basic Safety Messages (BSMs) used for vehicle-to-vehicle (V2V) and vehicle-to-infrastructure(V2I) communication. To simplify the analysis, only message contents that are related to vehicle movement and traffic events are considered. For example, the content of a message m_i^t can be represented as m^t, e^t where m^t represents the time that the message is generated, $s_t = \{x_t, v_t, a_t ...\}$ represents the vehicle state at time t including position, speed, acceleration, etc., e_t represents internal (e.g., emergency braking) or external events (e.g., road congestion, incidents, icy road, and weather conditions, etc.). Therefore, a V2V message broadcasted by a vehicle v_i can be denoted as $pk_i|m_i^t|\sigma(m_i^t)$ while a V2I message as $pk_i|e_i^t|\sigma(e_i^t)$.

An adversary can tamper with or spoof either vehicle movement or events of message contents. Fig. 5.2 gives time-space view [142] of an adversary spoofing and tampering with vehicle positions and movement before broadcasting them in V2X messages. Obviously, the forged vehicle positions (blue lines) seems to be implausible or follow a strange (e.g., zig-zag) trajectory.

There are four trajectory patterns in these forged vehicle positions [142]. Specifi-



(a) Spoofing stationary positions

(b) Adding constant offset to locations

(c) Spoofing random positions

(d) Adding random offset to locations

Figure 5-2: Time-space views of malicious behaviors: position spoofing [142]

cally, an adversary can have a compromised vehicle broadcast stationary (Fig. 5-2a), random positions (Fig. 5-2c), and add constant (Fig. 5-2b) or random offset (Fig. 5-2d) to actual positions of the compromised vehicle. These heuristics becomes useful when developing rules to filter out suspicious V2X messages.

We define two types of entities involved in V2X trust management-a *claimer* and a *verifier*. A *claimer* is a vehicle who broadcasts V2X messages with its own status or observed traffic events. A *verifier* can be a vehicle under vehicle-based trust management or a RSU under infrastructure-based trust management. They both share the evaluations of the vehicle's trustworthiness to the TA, which will revoke the certificate assigned to the vehicle if its accumulated reputation score is lower than a

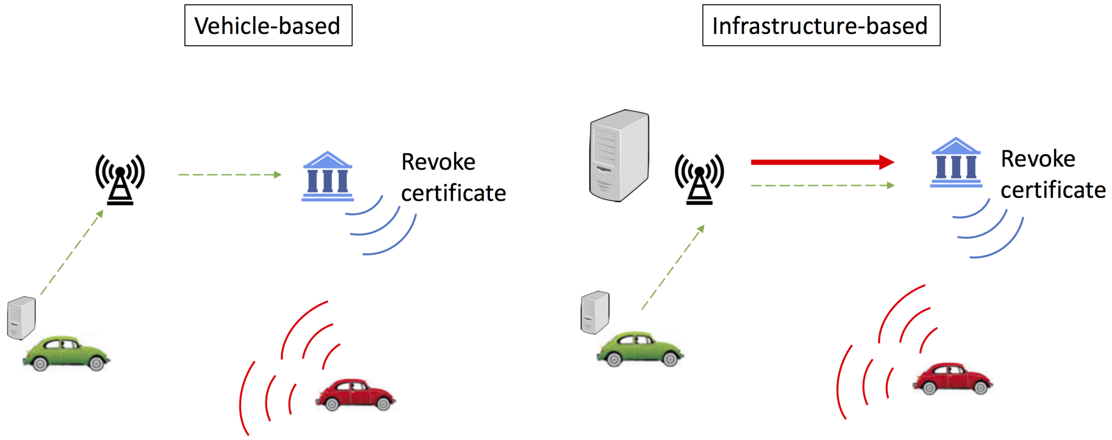


Figure 5-3: Two types of architectures for trust management

predefined threshold.

This chapter will explore two types of architectures for trust management, as shown in 5-3. In the vehicle-based architecture, each vehicle (as a verifier) will generate its evaluations on the trustworthiness (e.g., represented by a reputation score) of the vehicle with which it communicates through V2V message exchange. Each verifier vehicle will then share a reputation score of the target vehicle to the trust authority. The infrastructure-based approach, on the other hand, relies on trust evaluations by RSUs (as verifiers). more importantly, it is for evaluating the trustworthiness of V2I messages and the message origins, rather than V2V messages.

5.3 Trust management based on vehicle malicious reports

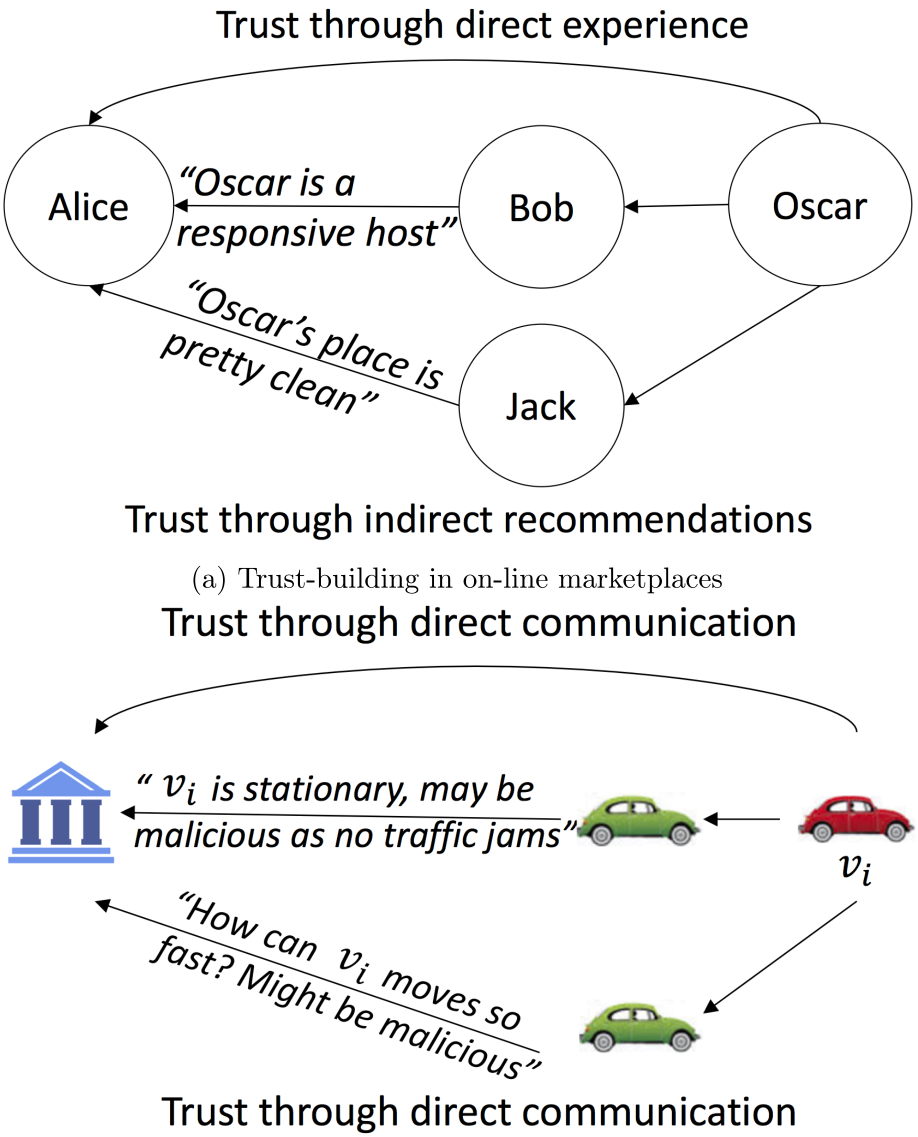
5.3.1 Motivation

As discussed in 5.1, a trust authority needs to determine the trust worthiness of a vehicle in order to decide whether to revoke the certificate or credential assigned to it. Existing principles that govern trust establishment in social networks can be extended to vehicular networks for trust management. Fig 5-4 provides an analogy between these two types of networks.

- Paths for trust propagation: In human society, a person can build trust on a given person through either direct interaction between them or based on indirectly recommendations from other persons, as shown in Fig. 5-4a [55]. For example, If a customer Alice chooses her host for lodging in on-line market places such as airbnb, she can build her trust on a given host Oscar either through her own experience of staying or get recommendations from Bob and Jack who have already stayed with Oscar before (Fig. 5-4a). Similarly, a TA can build trust on a given vehicle v_i through direct communication or from indirect recommendations from vehicle verifiers (Fig. 5-4b).

- Aggregation of opinions from different aspects: as Sabater and Sierra suggest, reputation of a person or an entity can be evaluated from different social dimensions [114]. Take again the example of on-line market place of lodging. Alice may evaluate the reputation of the host Oscar from multiple angles including the rate, the cleanliness and the comfort of the room, whether the host is responsive and polite, etc. as shown in Fig. 5-4a. Similarly, in vehicular networks, a verifier can evaluate the trust of the vehicle v_i by checking the plausibility of v_i's trajectory by using different rules [66, 120]. While one rule indicates malicious behaviors because it is implausible that any vehicle may always stay in the same place or move slowly even under smooth traffic flow, another rule may detect vehicles moving at a fast speed as malicious due to speed limits. In this regard, each module for plausibility checks contributes different weights to the evaluation of trust values on a given vehicle depending on the specific contexts.

- Roles and reputations of intermediate nodes: in the human society, people tend to trust a person who is recommended by someone who has good reputation. Similarly, a (vehicle) verifier who enjoys higher trust value (reputation) in the trust authority will have greater influence on the process of trust building. This aspect is reflected in Equation (1) proposed by Zacharia for calculating and updating trust values [155].


Trust through direct experience
Alice
"Oscar is a responsive host"
Bob
Oscar
"Oscar's place is pretty clean"
Jack
Trust through indirect recommendations
Trust through direct communication
" v_i is stationary, may be malicious as no traffic jams"
v_i
"How can v_i moves so fast? Might be malicious"
Trust through direct communication

(a) Trust-building in on-line marketplaces

(b) Trust-building in vehicular networks

Figure 5-4: An analogy of trust-building between social and vehicular networks [129]

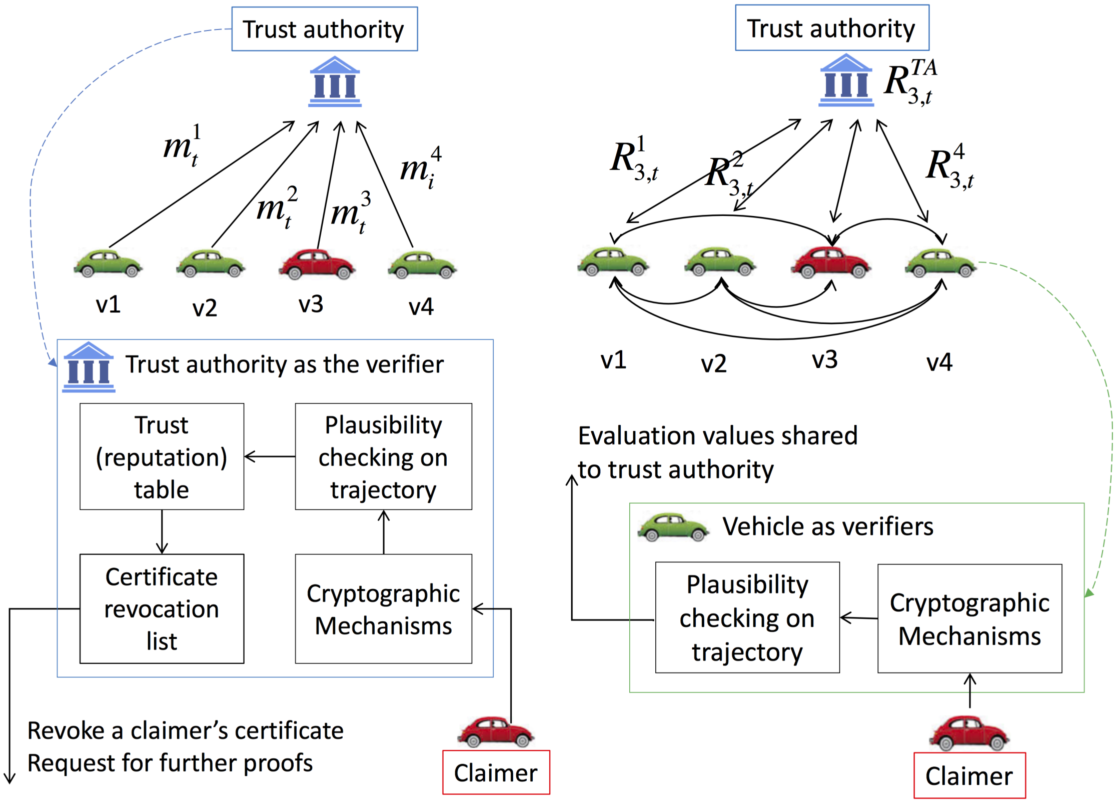
5.3.2 Trust architectures

For vehicle-based schemes, there are two types of architectures for vehicle trust management derived from principles in social networks. Design rationales for the centralized and distributed architectures are also discussed. In general, in the centralized architecture, the TA takes the role of verifier that directly checks V2V messages for building trust, as shown in Fig. 5-5a. On the other hand, the TA delegates the task of plausibility checking to vehicles in the distributed architecture, as shown in Fig. 5-5b.

In the latter case, vehicles become actual verifiers who send their evaluation of trust on nearby vehicles to the TA.

In a centralized architecture, the TA monitors vehicle operation and interactions through V2X messages within a pre-determined region. V2X messages broadcasted by each vehicle is collected and then sent to the trust authority for verification (Fig. 5-5a upper-left). It should be addressed though that we ignore the infrastructure such as roadside units (RSUs) or digital sensors deployed on the road for collecting the V2X messages send by vehicles. In such design, the TA serves the role of authenticating vehicle identity and message integrity, checking the plausibility of the content of messages and calculating trust on the origin of a given message, as illustrated in Fig. 5-5a (bottom-left). The advantage of centralized architecture is that the TA has a broader view of each vehicle's behaviors in terms of time duration and distance of travels, which is useful in detecting malicious behaviors globally. However, this type of designs comes with the price of increased memory consumption of storing trajectory and computation overhead for conducting plausibility checks on each vehicle.

In a distributed architecture, the responsibility of verifying the plausibility of vehicle trajectory is delegated to vehicles. Fig. 5-5b shows the detailed process of this. After evaluating the plausibility of claimed vehicle position or trajectory, a vehicle verifier will determine the trust value on the claimed vehicle and share the results to the trust authority. For this reason, the trust authority will only need to update its trust on the corresponding node. In addition to reducing memory consumption and computation overhead of the trust authority, another reason for adopting this design is to add more detection capabilities against local malicious behaviors such as spoofing a position that is out of the signal transmission range of vehicles [65]. However, the vehicle verifier may not be capable of detecting attackers in a global scale as the TA in the centralized architecture due to the ephemeral nature of vehicular networks. As Gerlach suggests, two vehicles that meet at current timestamp are not guaranteed to meet at the next [40]. In addition, distributed vehicles suffer the potential of over-trust and is more subjected to Sybil attacks, which will be discussed in the results of experimental simulations.


Trust authority
m_t^1
m_t^2
m_t^3
m_i^4
v1
v2
v3
v4
Trust authority as the verifier
Trust (reputation) table
Plausibility checking on trajectory
Certificate revocation list
Cryptographic Mechanisms
Revoke a claimer's certificate
Request for further proofs
Claimer
Trust authority
$R_{3,t}^{TA}$
$R_{3,t}^1$
$R_{3,t}^2$
$R_{3,t}^4$
v1
v2
v3
v4
Evaluation values shared to trust authority
Vehicle as verifiers
Plausibility checking on trajectory
Cryptographic Mechanisms
Claimer

(a) The centralized architecture

(b) The distributed architecture

Figure 5-5: Two proposed architectures for trust-building [129]

5.3.3 Trust models

In both centralized and distributed architecture, the internal data structure and algorithms that a verifier, either a TA or a vehicle, uses are similar. We first describe the process in details of how the TA in a centralized architecture builds trust on a given vehicle and then summarize the difference in internal designs between centralized and distributed architecture.

Initialization

The TA maintains two data structures in its memory: a trajectory table $\{(v_i, trj(v_i)) : v_i \in V\}$ where V is the set of registered vehicle ID and $trj(v_i)$ represents the list of vehicle v_i's trajectory; a trust table $\{(v_i, trust(v_i)) : v_i \in V\}$ where $trust(v_i)$ denotes the trust history of v_i evaluated by TA. When a message (v_i, t, s_t, e_t) is received from a new vehicle, the TA will establish corresponding entries $trj(v_i)$ and $trust(v_i)$ for v_i

in the trajectory and trust tables respectively. v_i will be also given an initial trust value I.

Plausibility checking

If the identity of the source node that sends a given V2X message (v_i, t, s_t, e_t) can be authenticated, the verifier will extract the trajectory information of the source node v_i from its trajectory table $trj(v_i)$ for plausibility checks. The plausibility checking module within a verifier can be regarded as a black box that takes as inputs the current message (v_i, t, s_t, e_t) sent by v_i's and its trajectory $\{(n, x_n^i, y_n^i) : n = 1, 2, ...t - 1\}$ where x_n^i and y_n^i represent latitude and longitude coordinates of v_i at n^{th} time step, and outputs a value $T_{i,t}$ that represents the evaluation made by a verifier on the trust of v_i.

Designing plausibility checking for TA verifier in centralized architecture and a vehicle verifier in distributed architecture differs in three aspects: selecting detection algorithms, designing internal parameters for each algorithm, and deciding on the weight assigned to each algorithm (its contribution to the final evaluation score of trust on v_i).

Algorithms used by the TA in centralized architecture can detect v_i's malicious behaviors in a global scale because the TA maintains and has real-time access to all observed positions along the trajectory of a given vehicle v_i. For example, the TA can use tracking algorithms such as Kalman filter [139] to track v_i's positions along the route of its travel and compare the predicted position with the position claimed by v_i [7, 143]. Comparatively, a vehicle verifier in distributed architecture may not be capable of detecting malicious behaviors in a global scale. Due to the dynamic nature of vehicular networks, a vehicle verifer v_j can only accumulate limited trajectory information of v_i as "two vehicles that meet at current time stamp is not guaranteed to meet at the next." [40] However, v_j may rely on the properties of the (radio) transmission signal such as radio propagation model to detect malicious behaviors in a local scale [65, 154]. For example, v_i will be regarded as malicious if the position it claims is out of the transmission range (e.g., $<=$ 300m) of signals received by v_j. Multiple vehicle verifier can even collaborate to check position claims based on multilateration principle [17] although this algorithm requires the sharing of observed positions of claimer vehicle v_i among multiple verifiers.

Details of tuning internal variables for each algorithms are not the focus of this paper and can be found in [66, 65, 120, 143]. We also do not include a detailed discussion on how to determine weights for each algorithms although address that they can be changed dynamically depending on the specific context, as discussed in the analogy between social and vehicular networks.

Update trust

We adopt the model for updating trust value proposed by Zacharia [155] in his study of reputation systems for on-line community and make necessaries changes. Equation 5.3a-5.3c [155] are adapted to the needs of trust-building in centralized architectures while Equation 5.2 [155] to the needs of distributed architecture.

In the centralized architecture, after the new trust value of vehicle v_i is calculated based on results from plausibility checking, the final evaluation value $T^{TA}_{i,t}$ will be sent to the subcomponent for handling trust-building within TA, as shown in Fig. 5-5a (lower-left). It will then update its trust on v_i by using 5.3a. Equation 5.3a is adapted to the needs of trust-building in the centralized architecture. We keep the definition of parameters the same where θ represents the learning factor that decides how fast each claimer vehicle's trust in the TA changes after each iteration; Φ is called damping function which reflects the contribution to updated trust value on v_i from its previous trust rating; D represents the maximum trust value that a vehicle can get and is different from Zacharia's original definition because TA will fully trust his own evaluation of v_i's trustworthyness; $E^{TA}_{i,t-1}$ is the normalized trust value of v_i in previous time step which is defined in 5.3a. Equation 5.3b defines the ϕ function where σ, the "forgetting factor" [155], decides the level of influence by previous trust value $R^{TA}_{i,t-1}$ on updated one $R^{TA}_{i,t}$.

$$R^{TA}_{i,t} = R^{TA}_{i,t-1} + \frac{1}{\theta}\Phi(R^{TA}_{i,t-1})D(T^{TA}_{i,t} - E^{TA}_{i,t-1}) \tag{5.1a}$$

$$\Phi(R_{i,t-1}^{TA}) = 1 - \frac{1}{1 + \exp\left(\frac{-(R_{i,t-1}^{TA} - D)}{\sigma}\right)} \tag{5.1b}$$

$$E_{i,t-1}^{TA} = R_{i,t-1}^{TA}/D \tag{5.1c}$$

In the distributed architecture, any vehicle verifier, say v_j, can send its evaluation of trust on claimer vehicle v_i from v_j's internal plausibility checking, denoted as $T_{i,t}^{j}$, to TA. The TA will then update its trust on v_i by using 5.2, as shown in Fig. 5-5b (lower-right). Note 5.2 differs from 5.3a in that parameter D, the maximum range of trust value, is substituted with $R_{j,t-1}^{TA}$ that represents how much that TA trust verifier v_j who shares its trust evaluation on v_i. This corresponds to the third principles of trust propagation in social networks that an agent's own reputation may influence how trustworthy that its recommendations are.

$$R_{i,t}^{TA} = R_{i,t-1}^{TA} + \frac{1}{\theta}\Phi(R_{i,t-1}^{TA})R_{j,t-1}^{TA}(T_{i,t}^{j} - E_{i,t-1}^{TA}) \tag{5.2}$$

Revoke certificates

If vehicle v_i is detected to be malicious, the TA can choose to revoke the certificate assigned to it. Then the question for designing a trust-building scheme becomes: given the trust history of v_i, when and how the TA should revoke certificates? We provide three strategies to illustrate the notion of trust-based certificate revocation although actual designs and implementations are contingent on engineers' choice.

- Assign a neutral trust value (e.g., I = D/2) to v_i and revoke its certificate when v_i's trust drops below a certain threshold
- Assign a low trust score (e.g., I = 0 or D/10) to v_i and revoke v_i's certificate if its trust value start to decline at a certain point and the trend continues for a (pre-defined) period of time steps.
- In privacy-preserved scheme where every vehicle holds multiple short-lived certificates [158], mechanisms for checking the linkage between vehicles and its assigned certificates can be added to TA such that TA may revoke v_i directly.

Table 5.1: Parameter choice in simulation

Parameter name		**Values**
Parameters for trust calculation in (1)-(4)	θ	20
	σ	200
	D	3000
Parameters for algorithms in plausibility checking	Min dist. moved	5
	Max speed difference	25
	Interval for dist. checking	5
	Max signal trans. range	200
	Threshold for sudden appearance	200

5.3.4 Experimental evaluation

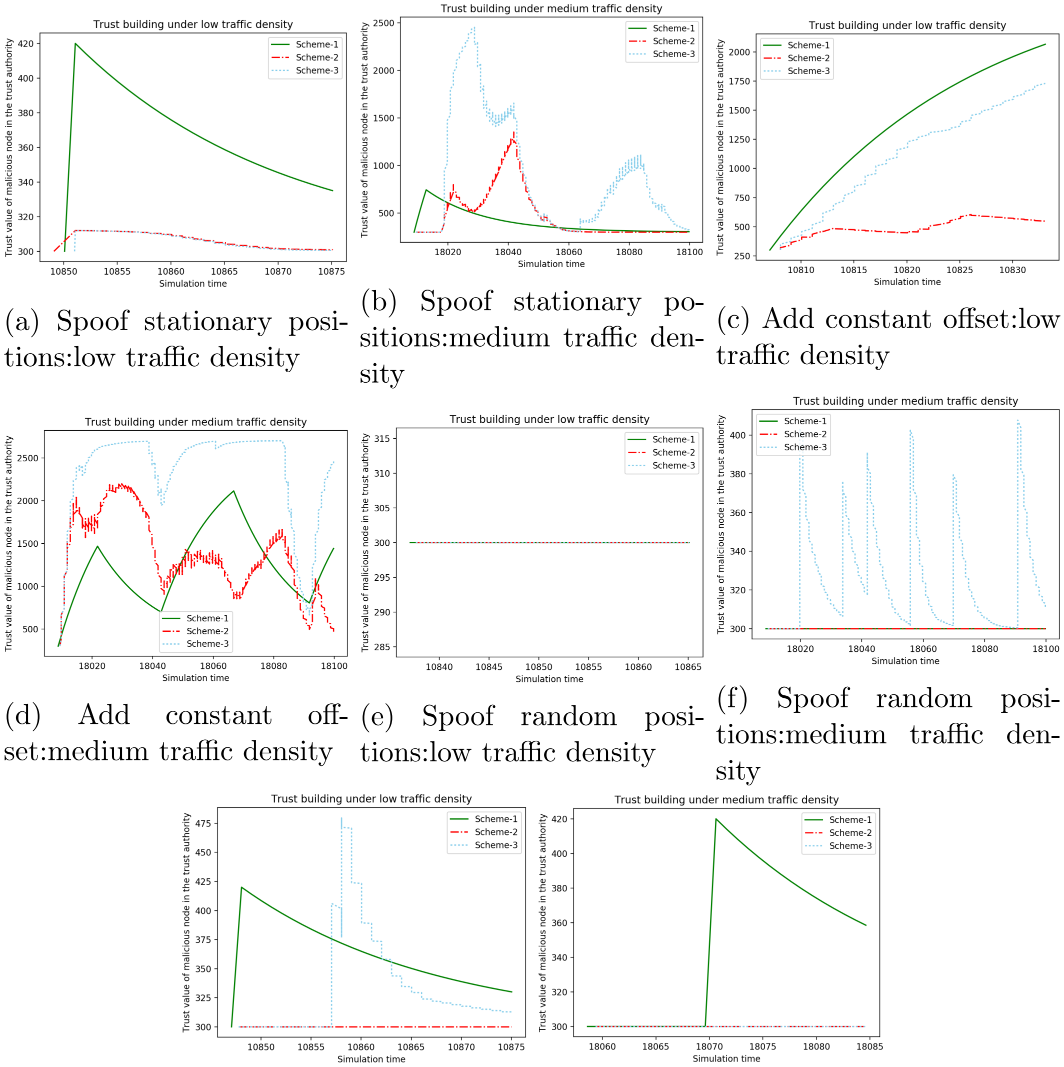
In this section, we evaluate two proposed architectures for trust-building on vehicle nodes by using simulation. We implement a discrete event simulator written in Python to emulate the process of V2V message exchange between different nodes. Specifically, for the trust-building module, we implement verifiers for TA in the centralized architecture and for vehicles in the distributed architectures based on the trust-building models discussed before. All parameters we used for calculating trust are summarized in Table 5.1.

For the module of plausibility checking, we utilize four types of existing algorithms that are developed in [120, 66, 143], rather than developing our own as our focus is to compare centralized and distributed architectures to inform the design of trust-based schemes. There are two algorithms that are used by both TA and vehicle verifiers including: *Minimum Distance Moved* (*MDM*) [66] for checking whether a given vehicle has moved a threshold distance during a pre-defined interval and *Simple Speed Check*(*SSC*) [143] which is a simplified Kalman Filter for checking the plausibility of vehicle movement based on estimation of maximum vehicle speed. Also, there are two algorithms we used are only for vehicle verifiers and they can only detect malicious behaviors locally: *Acceptance Range Threshold* (*ART*) [66] that can detect whether the claimed position of a given vehicle is out of the transmission range of signal (e.g., 300m) and *Sudden Appearance Warning* (*SAW*) [66] whether a given vehicle appears suddenly in front of a vehicle verifier, indicating suspicious behaviors. If any of these algorithms within a given verifier has detected malicious behaviors, the corresponding verifier will output a low trust rating (0.1 in our implementation). Similarly, a verifer will output a high trust rating (0.9 in our implementation) if none of the algorithms above detect any malicious behaviors. It should be addressed that this is a simplified implementation of the plausibility checking module discussed in this paper as each module is supposed to be assigned to a weight that decides the contribution to final trust rating respectively, all parameters we selected for implementation is summarized in Table 5.1.

In summary, there are three schemes that we evaluate in our python simulator and compare them based on four criteria. In particular, we take out of the two algorithms for detecting local malicious behaviors from the verifers as a benchmark for comparing centralized and distributed architecture.

- Scheme-1: Centralized architecture.
- Scheme-2: Distributed architecture with local detection algorithms.
- Scheme-3: Distributed architecture without local detection algorithms (i.e., ART and SAW).

The input dataset to our simulator is an open-source dataset-VeReMi-for evaluating algorithms for detecting malicious behaviors in vehicular networks [142]. VeReMi implements four types of malicious behaviors where attackers periodically or randomly sending malicious messages. This is illustrated with four examples shown in Fig. 5-2: broadcasting stationary positions (Fig. 5-2a), adding constant offset to actual positions of the claimer vehicle (Fig. 5-2b), broadcasting random positions in a given region (Fig. 5-2c), and adding random offset to actual position of the claimer vehicle (Fig. 5-2d). Note the attacker's behaviors appear to be less aggressive in the first two cases than in the last two cases in that the attacker changed claimed positions more abruptly and frequently in a short time duration. We choose data files corresponding to each of these malicious behaviors with two levels of traffic density



(a) Spoof stationary positions:low traffic density

(b) Spoof stationary positions:medium traffic density

(c) Add constant offset:low traffic density

(d) Add constant offset:medium traffic density

(e) Spoof random positions:low traffic density

(f) Spoof random positions:medium traffic density

(g) Add random offset:low traffic density

(h) Add random offset:medium traffic density

Figure 5-6: Simulation results: real-time trust value of malicious vehicle in the trust authority [129]

(i.e., low with 35 vehicles and medium with 97 vehicles in total), which amounts to 8 rounds of simulations. The simulation time is approximately 25 seconds for low density and 100 seconds for medium density conditions.

5.3.5 Results and discussion

Both centralized and distributed architectures are tested against four types of malicious behaviors under low and medium density conditions, as shown in Fig. 5-6. All X axes in Fig. 5-6a-5-6h represent the elapsed time since the start of the simulation (we only show partial time window) and y axes represent the TA's perceived trustworthiness of the target vehicle (a malicious node). In a nutshell, scheme-2 (denoted by red lines) and scheme-3 (denoted by blue lines) each of which is a type of distributed architecture and should've performed better than scheme-1 (denoted by green lines) by intuition in all scenarios because of more on-board detection capabilities, suffer "over-trust" for medium traffic conditions. In other words, the malicious node is able to indirectly get a high level of trust by the TA during short time interval by first poisoning other vehicles it interacts with.

Capabilities of maintaining low trust for attacker nodes

The response capability we discuss here involves two aspects: first, how fast the TA can adjust its trust on a given vehicle when the vehicle starts to behave maliciously; second, whether the TA is able to keep the trust value of the malicious vehicles at a low level given different strategies an adversary take (e.g., sending spoof messages intermittently or gradually adding small perturbations to actual positions).

From the evaluation results, we can see that scheme-2 outperforms the first two schemes in almost all scenarios except under medium traffic density conditions (Fig. 5-6b and 5-6d), the trust value of attacker nodes in scheme-2 are sometimes greater than in scheme-1. Not only does the TA can respond swiftly to malicious behaviors, but also maintain low trust scores for malicious vehicles. Attacker nodes fail to build up their reputations in the trust authority in Fig. 5-6a, 5-6c, 5-6f, 5-6g, and 5-6h. One reason may be that the vehicle verifiers in distributed architecture can detect malicious patterns in a local scale that are not visible to the trust authority in centralized architecture. One example is that local vehicle verifiers can decide if range between their position and the claimed position by the malicious node is greater than the transmission range threshold, as discussed before.

Vulnerability of distributed architecture in building trust

The poor performance of scheme-2 in Fig. 5-6b and 5-6d can be attributed to the ephemeral nature of vehicular networks. Indeed, it is not guaranteed that a vehicle (local) verifier in the distributed architecture can collect enough trajectory information of a given vehicle to determine its trustworthiness. Consider the scenarios that a malicious vehicle broadcasts (fake) stationary positions intermittently: it always broadcasts the same position for a couple of seconds (less than the detection threshold), wait for a while, and repeat again. If this happens, vehicle verifies in scheme-2 and 3 (distributed architecture) will not be able to respond to malicious behaviors. The same as malicious vehicles that add constant offset to their actual positions, as shown in Fig. 5-6b and 5-6d. On the other hand, scheme-1 performs better in attack scenarios with medium density traffic because the TA has a "global view" of vehicles' trajectory. Another possible explanation may be "over-trust" to which distributed architecture is vulnerable. Consider Fig. 5-6b and 5-6d again. The higher reputation of attacker nodes under scheme-2 (compared to scheme-1) may be because attackers first send regular messages to make (local) vehicle verifiers believe they are benign ones.

Memory, computation and communication overhead

Since the memory consumption and computation overhead incurred by trust calculation are negligible to storing and checking plausibility of vehicle trajectory, we only consider the latter one. We make three assumptions to facilitate the analysis: first, there are n vehicles in the region monitored by the TA; second, each verifier, regardless of whether it is the TA or a vehicle, only keeps the most m recent trajectory information for each vehicle it interacts with; third, the average running time of each algorithm for plausibility checking is $O(m)$. The third assumption is reasonable as the maximum time that a plausibility-checking algorithm takes to finish calculation corresponds to the length of the trajectory table. Furthermore, all algorithms for plausibility checking are running concurrently or the total number is much less than

m, we can assume that the total running time of the plausibility checking module is also $O(m)$.

For the TA in centralized architecture, the memory consumption to store the trajectory table for all vehicles is $O(mn)$. The total running time of plausibility checking on trajectory of all vehicles is $O(mn)$ for fully sequential processing and $O(m)$ for fully parallel processing in the ideal case where the TA is fully aware of the expected number of vehicles in the networks. On the other hand, the TA in the distributed architecture only needs to store and update the trust value of each vehicles while the computation of plausibility checking are performed by vehicle verifiers.

For the communication overhead, scheme-1, 2 and 3 will have the same size of certificates, keys and signatures. However, scheme-1 suffers the overhead generated by transmitting vehicles' trajectory to the TA, while each vehicle only needs to provide its trust evaluations for neighbours in scheme-2 and 3.

Privacy preserving property

Although privacy is not the focus of this paper, it is worth mentioning the potential conflict between security and privacy, which will be one of our future research directions. On one hand, a lot of design efforts in public key infrastructure is to add privacy-preserving properties to certificates [158], including shared, short-lived and group signature, etc. On the other hand, that TA conducts plausibility checking and builds trust for each vehicle may break unlinkability of V2X messages. In other words, an adversary can link multiple messages with the origin vehicle such that he or she can track the vehicle. This can happen if a verifier, either a TA or a vehicle, is compromised. Obviously, the distributed architecture requires more protection from privacy breach because each vehicle verifier needs to link multiple messages with a given vehicle that sends these messages in order to evaluate its trust.

5.4 Trust management based on infrastructure support: A Proof-of-Travel protocol

Vehicle-based trust management discussed in the previous section can mitigate the risk of adversaries tampering with and spoofing vehicle positions and movement contained in V2V messages. It also helps the TA determine the reputation of a vehicle based on malicious-behavior reports from surrounding vehicles. However, it cannot deal with forged V2I messages about traffic and road conditions. Besides, it doesn't answer the fundamental question regarding the adoption V2X communication: What are the social and economic factors that incentivize the owner of a vehicle, to participate in V2X communication and share its observations? We propose a Proof-of-Travel (POT) protocol to resolve these two issues:

- Determining the trustworthiness of V2I messages: Information about traffic and roads disseminated by V2I channels will play a crucial role in intelligent transportation systems in the near future. The transportation management center (TMC) can use V2I information [90] for real-time traffic management. For example, messages about the location of a work zone reported by a connected vehicle can be disseminated to other vehicles in the same region for re-routing and avoiding congestion. Similarly, V2I messages about the location and severity of an incident can assist the TMC in allocating resources of law enforcement, fire crews, and medical assistance to the accident site [91, 19]. Therefore, it is necessary to incorporate into the V2I infrastructure the mechanism for verifying the authenticity and determining the accuracy of the time, location, and severity of these V2I events before disseminating them [111, 27].

- Incentivizing stakeholders to opt-in for V2X-based services: Currently, the technology development for V2X services is still in the testing or concept stage, and participants in the testing activities are chosen on a voluntary basis [6]. What often occurs is that a traditional vehicle model needs to be installed with aftermarket OBUs supporting V2X communication to participate in the testing of connected vehicles. In the near future when V2X technologies become mature enough for mass deployment, the key challenge will become how to engage more customers who are willing to use V2X services to share their vehicle-collected information. Issues related to V2X adoption and deployment (including security) need to be considered in the social and economic contexts with a broader scope [81].

5.4.1 Protocol overview

The POT protocol is designed to help the TA and infrastructure components (e.g., RSUs) determine the trustworthiness and reward of vehicle nodes. Any vehicles which try to earn reputation scores must show valid proof of spatial movement testified by infrastructure components. The design rational is as follow: the requirement for spatial movement following a "meaningful" trajectory to build reputation creates extra burdens for a malicious node whose only objective is to compromise V2I communication by spoofing fake traffic events, but not for a normal vehicle which naturally wants to move from the origin to the destination.

Specifically, POT defines the V2I message format and the communication procedure for a vehicle to acquire location proofs, titled "location signature" formally defined later, from each trustable infrastructure component (e.g., RSUs) along its path of movement. The chain of proof hold by a vehicle testifies both the vehicle's claimed trajectory and its contributions to the transportation system, which forms the foundation for building incentive mechanisms for V2X services. This “dual role” aligns with the design goals of POT mentioned earlier: First, the prerequisite that the vehicle must be physically present at a given location to get the location proof from the corresponding RSU creates a burden of spatial movement and increase the cost of being malicious. Second, since the vehicle's altruistic behavior of sharing its status and observations of traffic events can now be formally verified (using cryptography techniques), the transportation system as a whole can form a consensus for measuring the contribution of the vehicle, which can be used to determine its reward.

We first give definitions necessary for the POT protocol and examine the com-

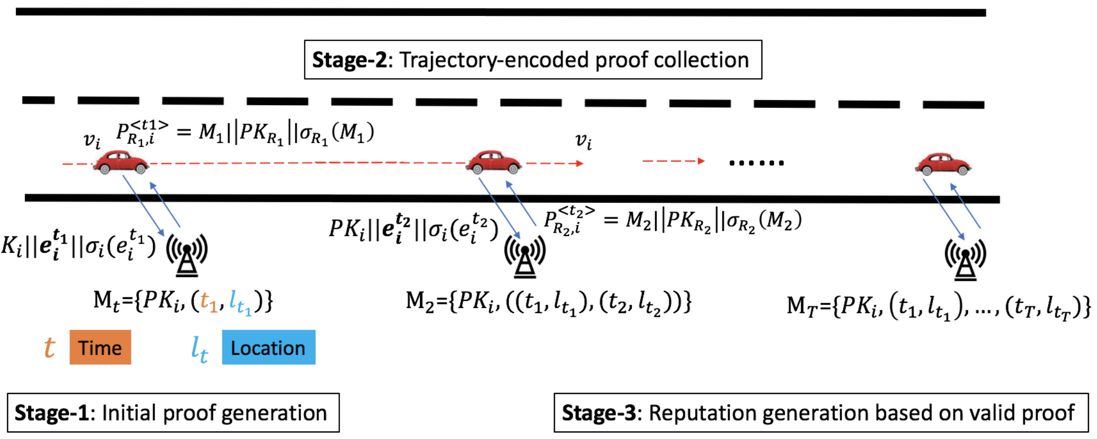

Stage-2: Trajectory-encoded proof collection
v_i $P_{R_1,i}^{<t1>} = M_1||PK_{R_1}||\sigma_{R_1}(M_1)$
v_i
......
$PK_i||e_i^{t_1}||\sigma_i(e_i^{t_1})$
$PK_i||e_i^{t_2}||\sigma_i(e_i^{t_2})$
$P_{R_2,i}^{<t_2>} = M_2||PK_{R_2}||\sigma_{R_2}(M_2)$
$M_t=\{PK_i,(t_1,l_{t_1})\}$
$M_2=\{PK_i,((t_1,l_{t_1}),(t_2,l_{t_2}))\}$
$M_T=\{PK_i,(t_1,l_{t_1}),\ldots,(t_T,l_{t_T})\}$
t Time
l_t Location
Stage-1: Initial proof generation
Stage-3: Reputation generation based on valid proof

Figure 5-7: The detailed process of the Proof-of-Travel protocol

munication procedure (5-7). The security analysis on the protocol is then presented. The effectiveness of the POT protocol for mitigating insider adversaries is evaluated by a case study on consensus formation on V2I events.

5.4.2 Preliminaries of the POT protocol

Definition 5.1 A *location signature* [24, 20, 95] issued by a RSU (denoted as rsu_j) to vehicle v_i at time $t = t_k$ is defined as

$$ls_{i,j}^{<t>} = pk_{rsu_j}||pk_{v_i}||t_k||h_e||h_{pre}||\sigma_{rsu_j}(pk_{rsu_j}||pk_{v_i}||t_k||h_e||h_{pre}),$$

where pk_{rsu_j} represents the public key of rsu_j, pk_{v_i} represents the public key of v_i, h_e denotes the hash of all event information reported by v_i, h_{pre} denotes the hash of the location signature v_i acquired from the previous RSU along its path of movement.

To construct a location signature, rsu_j signs on the contents above to generate the corresponding digital signature $sig_{rsu_j}^{tk}$, which guarantees the authenticity of the contents and legitimacy of the signature.

A location signature is a geo- and time-stamped message issued by an RSU to a particular vehicle to attest the vehicle's presence in a particular location at a given time. Since the contents signed by the RSU contain information about the vehicle's observations of traffic events and road conditions (i.e., the hash of vehicle reported events h_e), the RSU also "admits," on behalf of other trust entities, the contributions made by the vehicle as the shared information can be used by other vehicles or by the TMC, as mentioned earlier.

Definition 5.2 Proof of Travel for vehicle v_i is the set of location signatures = $ls_{i,j}^{t_0}, ls_{i,j+1}^{t_1}, ..., ls_{i,j+T}^{t_T}$ that v_i acquired from RSUs along the path of its movement during the time interval T.

Definition 5.3 Verifiable vehicle miles traveled (VVMT) for vehicle v_i is denoted by
$vvmt_i^{<t_k,t_{k+1}>} = d(ls^{t_k}, ls^{t_{k+1}})$ and defined to be the distance v_i has moved between two locations each of which corresponds to a location signature v_i acquired at time t_k and t_{k+1} respectively.

We assume that the location of each RSU is predetermined and fixed. Therefore, other RSUs and the TA can derive the exact location of a given RSU from its public key included in the location signature.

Any vehicle node which wants to be eligible for participating in V2X activities such as reporting traffic events for building reputation and gaining reward must at least accumulate a predetermined level of VVMT.

5.4.3 Detailed communication procedure for Proof-of-Travel

A vehicle following the POT protocol will start to acquire and accumulate proofs from RSUs after it joins the vehicular network, and the process continues until it exits the road with V2X coverage. The reputation and reward of each vehicle participating on V2I communication activities (indicated by VVMT) is then determined by the length of the chain of valid proofs the vehicle collects. The POT protocol consists of three stages, as presented below.

Stage-1: Initial proof generation

The proof-collection process begins with a vehicle (v_i) sending a location-signature request $req_{v_i}^{t_1} = pk_{v_i}||t_1||e_i^{t_1}||pos_i^{t_1}||\sigma_{v_i}(pk_{v_i}||t_1||e_i^{t_1}||pos_i^{t_1})$ to the first RSU (rsu_1) it meets after joining the vehicular network, as shown in Fig. 5-7. The request consists of the identity and authentication information of v_i, such as its encoded public key pk_{v_i} and the digital signature signed on this request $\sigma_{v_i}(.)$, and the observed traffic events $e_i^{t_1}$ or its own movement, such as its real-time position $pos_i^{t_1}$, speed $spd_i^{t_1}$, and acceleration $acl_i^{t_1}$, etc.

After receiving the request, rsu_1 will authenticate v_i's identity and the integrity of the request message by using the attached digital signature $\sigma_{v_i}(.)$. It may also check the plausibility of the information included in the request, such as $e_i^{t_1}$ or $pos_i^{t_1}$, by using pre-defined heuristic rules. For example, if the location of the event (e.g., work zone) reported by the vehicle is far away from the vehicle's own location, or the work zone is impossible to pass along the vehicle path, rsu_1 may reject the request.

If results from all the identity, integrity, and rule-based plausibility checks are judged to be valid, rsu_1 will generate a location signature ls_{v_i,rsu_1}^{t1} and send it to v_i.

State-2: Trajectory-encoded proof collection

When vehicle v_i meets the next RSU rsu_2, it will attach the location signature ls_{v_i,rsu_1}^{t1} acquired from rsu_1 when sending a new request $req_{v_i}^{t_2}$, as shown in Fig. 5-7. Similarly, in addition to verifying the new request based on the heuristics described earlier, rsu_2 will also check if the previous location signature ls_{v_i,rsu_1}^{t1} is owned by v_i, the vehicle sending the request (ownership checks), has not expired (time of validity), and was issued by a legitimate RSU (legitimacy checks), such as a valid RSU who has registered with the trust authority and is adjacent to or near rsu_2.

If all checks are valid, rsu_2 will construct a new location signature ls_{v_i,rsu_2}^{t2}. However, other than concatenating all the elements as discussed earlier, rsu_2 will also attach the hash value of the previous location signature ls_{v_i,rsu_1}^{t1}, sign on the merged data, and send the newly constructed location signature back to v_i. This process repeats until the vehicle has collected enough location signatures to form a chain of proofs, which are geo-time-stamped ledgers of v_i's trajectory history verified by all RSUs along the path of movement.

Stage-3: Vehicle trustworthiness based on location proofs

To determine the reputation (and the reward) of vehicle v_i, the TA or RSUs can use the chain of proofs owned by v_i to calculate its VVMT. This step differentiates the POT protocol from previous work that also utilizes RSU-authenticated vehicle trajectory [24, 20]. v_i's VVMT can be derived from all the location signatures included in the chain of proofs it collected and presents to the TA.

Before deriving the VVMT for the vehicle, the TA needs to check if all location signatures included in a chain of proofs indicate a plausible trajectory. For example, multiple location signatures owned by a vehicle may indicate an extremely fast speed impossible to achieve under the current traffic and road conditions. Also, Location signatures may form a strange trajectory (e.g., taking a zigzag line even if a straight line is the optimal route), which will reduce the likelihood of previous proofs being valid. The rules for verifying proofs should support fault tolerance in the case where v_i fails to get the location signatures from two adjacent RSUs due to faults or congestion in communication links. For example, a threshold signature scheme can be used in authenticating vehicle trajectory [8] such that only a subset (m) of all RSUs' signatures (n, m<n) is needed for determining the legitimacy of location proofs presented by a vehicle.

As mentioned in the introduction, VVMT can be used to calculate the reputation score of v_i. Although the choices of the exact formula for deriving the reputation score of a vehicle and the reward functions based on VVMT is out of the scope this paper, the overarching rule is that vehicles with higher VVMT will be viewed as more trustworthy (e.g., higher reputation score) and enjoy more rewards.

$$R_i = \sum_{1}^{T} R_i^{<t>} \tag{5.3a}$$

$$R_i^{<t>} = f(vvmt_i^t) = \alpha * vvmt_i^t \tag{5.3b}$$

The equation we use for illustrating how to determine the reputation of a given vehicle v_i based on POT protocol is given in equation 5.3, where R_i represents the total reputation score of v_i from the view of the TA, infrastructure components, and other vehicles, $R_i^{<t>}$ is the reputation score gained by "traveling" between the time interval t and $t-1$, and α is a constant that decides the linear relation between VVMT and its accumulated reputation perceived by other entities.

5.4.4 Security analysis

Since the communication between vehicles and road infrastructure is vulnerable to eavesdropping, colluding, and insider attacks, we need to evaluate whether and how POT protocol can defend against different types of attacks on V2I channels.

Replay attacks

Similar to other trajectory-based authentication approaches [24, 20], the POT protocol can prevent the misuse of location signatures intercepted by an adversary when eavesdropping V2I channels. This is because the public key of a vehicle attached to the location signature can ensure that only the vehicle who holds the corresponding private key can claim the "ownership" of the location signature and use it as proof.

Proof or trajectory forgery by "inside" adversaries

Incorporating cryptography hashing (e.g., sha256) into the signed data can prevent the forgery of chains of proofs. Since we have added the hash of the previous location signature into the contents of the current location signature during stage-2 of the POT protocol, any changes made to a particular location signature will also change its hash value, which results in an inconsistency between the hash of that particular location signature and the pre-hash value contained in the next location signature in the chain of proofs. This also means that an adversary who holds valid vehicle credentials and wants to forge a valid chain of proofs containing multiple location signatures must be physically present in each corresponding RSU. The adversary must also follow a “plausible” trajectory to gain VVMT as any RSU only accepts a location signature request if the previous signature attached to it is signed by another legitimate RSU (legitimacy checks on location signature in Stage-2), and the trust authority will verify the plausibility of a vehicle's claimed trajectory indicated by its chain of proofs (trajectory plausibility checks in Stage-3).

It is the cost of "compulsory" spatial movement for gaining location signatures that reduces the adversary's incentive for being malicious. However, the requirement for spatial movement will not incur extra cost to normal travelers.

Private key swapping by colluding nodes

Although POT does not completely eliminate colluding nodes, it can diminish the negative effect of misusing location signatures for the same reasons discussed in proof-forgery attacks. For example, even if a malicious vehicle node may tunnel the location signature it collects along with its private key to another colluding node, the former one must also share a valid chain of proofs it acquired for the latter node to gain VVMT. Sharing only a subset of locations signatures or modifying any location signatures will invalidate the whole chain of proofs. This also means that, from an economic perspective, the group of colluding nodes as a whole must always "pay the cost" incurred by spatial movement. When the cost becomes greater than the benefit the colluding nodes earn, they lose the incentive for forging V2I events.

5.5 Evaluation through a case study on consensus formation

To evaluate the effectiveness of the POT protocol, it is applied to the problem of consensus formation on the correctness of traffic events that vehicles in a local region (e.g., intersection) report. Safety-critical events are selected since the focus since transportation management and emergency response decisions are decided based such event reports, as shown in Fig. 5-1. An RSU, after receiving reports for the same event from multiple vehicles, will determine its correctness and forward this message to transportation agencies and law enforcement departments to request emergency response, as discussed in Chapter 3.

5.5.1 Previous work on consensus algorithms

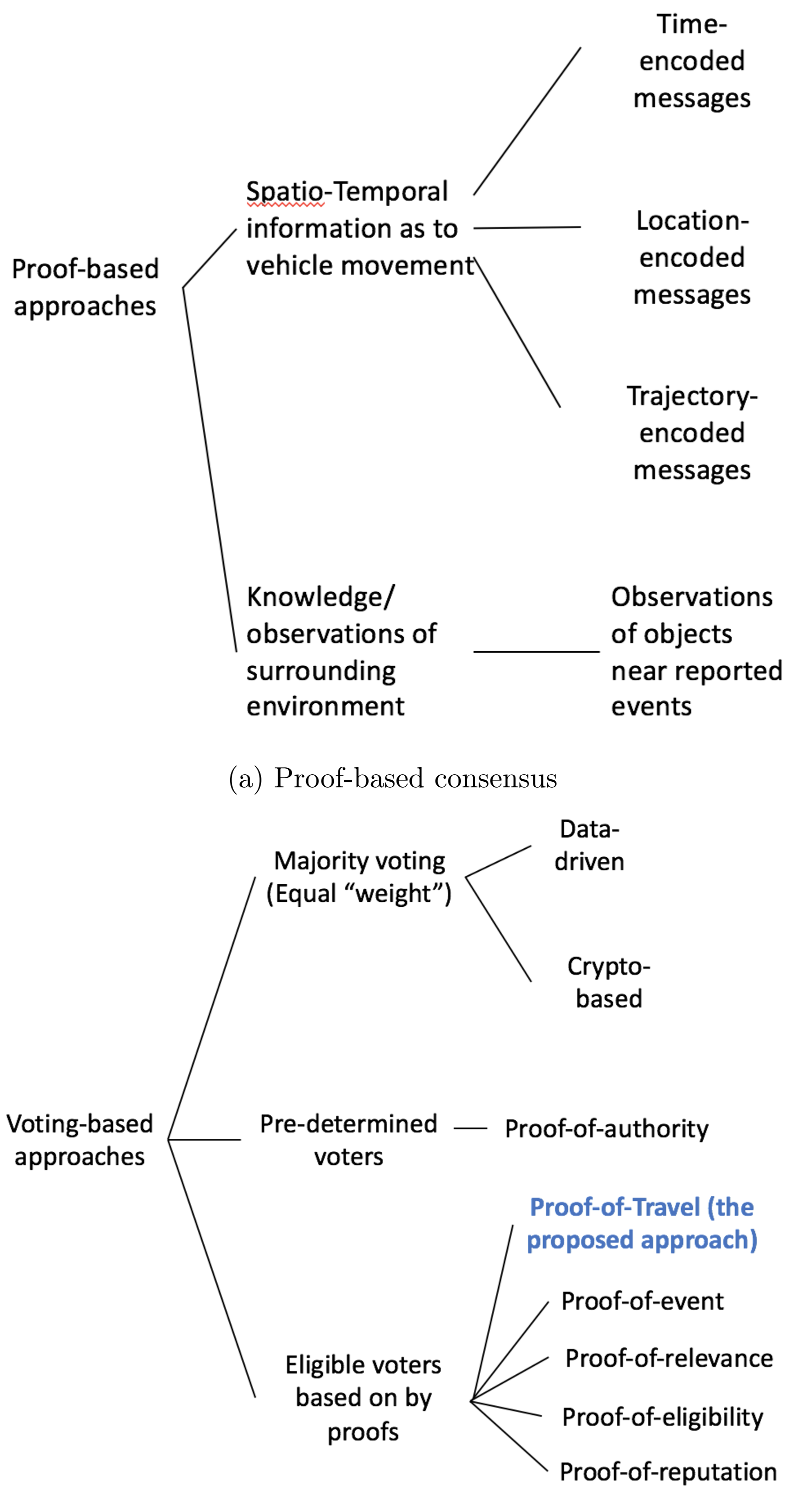
Consensus formation algorithms by vehicles and infrastructure can be classified into two categories, including proof-based and voting based. Fig. 5-8 provides a summary of previous works on consensus algorithms. The proposed POT-based voting in this section is a hybrid approach in the sense that the eligibility for a node to "vote" is determined by whether it can present enough "proof".

Proof-based consensus

Proof-based algorithms such as proof-of-work (POW) [82], proof-of-stake (POS) [63], and Proof-of-authority (POA) [33] have been developed for distributed systems.

In the context of vehicular networks supporting V2X communication, a proof is the context information that a claimer vehicle encodes in the V2I message it sends out. This information is used by verifiers (e.g., RSUs, the traffic controller on the cloud, or the trust authority) to verify message authenticity and integrity. Such designs are motivated by the fact that even vehicles with valid credentials can not be trusted due to the insider potential to be malicious. Therefore, redundant authentication mechanisms such as proofs are needed by verifers to detect insider attacks.

Two types of proof-based methods for forming consensus on events are presented: spatiotemporal information as proofs and knowledge/observations about the surrounding environment as proofs, as shown in Fig. 5-8a. The design process of proof-based methods illustrates the prioritization balance between security and privacy. There are two scenarios of insider adversaries which proof-based methods target. The first one is caused by the adoption of privacy-preserved authentication technologies such as group signature [22] or short-lived pseudonyms [158], which can be detected by encoding location and timing information. The second one is due to the physical access to OBUs that store vehicle credentials, which can be mitigated by encoding trajectory information.


Proof-based approaches
Spatio-Temporal information as to vehicle movement
Time-encoded messages
Location-encoded messages
Trajectory-encoded messages
Knowledge/ observations of surrounding environment
Observations of objects near reported events
Voting-based approaches
Majority voting (Equal “weight”)
Data-driven
Crypto-based
Pre-determined voters
Proof-of-authority
Eligible voters based on by proofs
Proof-of-Travel (the proposed approach)
Proof-of-event
Proof-of-relevance
Proof-of-eligibility
Proof-of-reputation

(a) Proof-based consensus

(b) Voting-based consensus

Figure 5-8: The categorization of consensus

a. Location and time-encoded proofs We focus on encoding location and time information to detect insider adversaries due to the adoption of group signature [22, 158] to protect vehicle privacy. The idea behind group signature is that each vehicle, as a member of a vehicle group, can generate a signature on behalf of the group and each group member can verify the signature without knowing the signer of the message. For this reason, it is possible that a malicious CAV may initiate denial-of-service (DOS) attacks by sending a large quantity of bogus and fake events to nearby vehicles or infrastructure components within a short time period without being detected.

For an infrastructure verifier to detect a DOS attack by using time-encoded V2I messages, it can require the claimer to sign on the concatenation of the message and the current time, rather than the raw messages, when generating digital signatures. If the verifier receives multiple V2I messages from a claimer vehicle regarding an event report, service request, or registration request (requesting a new session key), the verifier can check if the time interval between two consecutive requests from the same claimer is less than a pre-defined threshold [124, 74]. A violation of this rule might indicate a potential attack.

Similarly, location-encoded V2I messages can be used to deal with insiders in anonymous message authentication. For example, rather than use group signature or short-life pseudonyms for maintaining anonymity during V2I communication, existing authentication algorithms such as elliptic curve digital signature algorithm (ECDSA) can be enhanced such that a claimer vehicle can use a session key [13] that is generated by encoding its real-time locations to interact with verifiers in close proximity. Only the claimer vehicle with a session key indicating its close proximity to the RSU can pass the authentication. This can prevent an insider who tries to achieve repudiation (i.e., deny a previous behavior) by replaying old messages sent out by other vehicles.

b. Trajectory-encoded proofs The idea of using timing and location information of a claimer vehicle to generate proofs for identity authentication can be extended to vehicle trajectory for the duration of its movement [147]. The main use of trajectory-based authentication is to defend against Sybil nodes created by insider adversaries who hold a valid vehicle credential through physical access to OBUs, which is different from insider scenarios caused by the use of anonymous authentication schemes above.

The main idea behind a trajectory-based approach is the notion of similarity testing: Each vehicle has its unique movement pattern and the event that two vehicles pass multiple RSUs at the exact same time points along their trajectory is rare [20, 94]. To get the trajectory of a given claimer vehicle, a verifier can rely on either surrounding vehicles or infrastructure.

The former involves having any vehicles that the claimer vehicle meets along its trajectory serve as observers that receive anonymous beaconing from the claimer vehicle, and reports their "observations" to the verifier as proofs [75]. However, this approach assumes a high density of vehicles that support vehicular networks. On the other hand, infrastructure-based methods rely on RSUs deployed to road segments or intersection to generate location signatures as proofs after receiving requests from the claiming vehicle [20], which works for initial stages of connected vehicle deployment.

c. Knowledge and observations as proofs In addition to encoding spatio-temporal information in V2X messages, the knowledge that a CAV has about the surrounding environment such as visual clues made by on-board sensors can also be used in verifying essages. One example is the verifier will pose a challenge to test a claimer vehicle's knowledge about the color, type, or size of the vehicle involved in a traffic incident when verifying the incident report from that claimer [70].

Voting-based consensus

Voting-based consensus formation algorithms have been researched extensively in distributed computer systems. Numerous algorithms have been proposed to verify the authenticity and validity of messages shared by nodes and propagated through networks, such as original Byzantine fault tolerance [64], practical Byzantine fault tolerance (PBFT) [18], and speculative Byzantine fault tolerance [62].

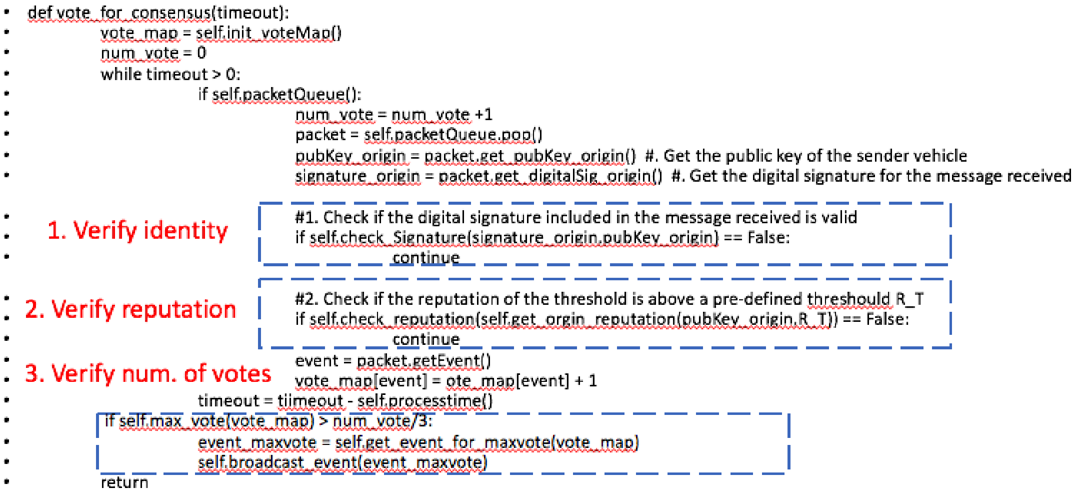
These works enable security engineers to look at the issue of V2I message authenticity and integrity through the lens of majority voting mechanisms. Voting-based methods often require multiple verifiers that the claimer vehicle meets along its path to collaborate in forming consensus on the vehicle's claimed locations or reported events.

Which entities are eligible to vote can be decided by using methods shown in Fig. 5-8b. The first way is that all vehicles in close proximity of the claimer vehicle can serve the role of the verifier and thus have equal weight in voting [154, 59]. The second way is to pre-define a set of trusted authorities (e.g., RSUs) that can vote for other entities. Similar ideas have been explored in blockchain-based transactions such as the POA [76] consensus protocol. The third way of selecting voters is a hybrid approach combining voting-based and proof-based and approaches. The legality of a potential voter is decided by whether it can present valid proofs of its presence in certain locations or observations along the trajectory path. Multiple schemes are proposed including proof-of-event [151], proof-of-relevance [16], and proof-of-eligibility [70].

5.5.2 Consensus formation based on Proof-of-Travel

The idea of using a POT protocol in consensus formation is to determine whether a certain vehicle is eligible to "vote" (i.e., participate in reporting a safety-critical event) based on its reputation accumulated throughout vehicle movement. To put it formally, for v_i to participate in voting for event A (e.g., whether A has occurred at a given location at a certain point of time), v_i must have reputation $R_i > R_T$ where R_T is the pre-determined threshold for voting. The voting algorithm that a RSU can take for forming consensus on the correctness of an event is given in Fig. 5-9.

The voting algorithm adopted by RSUs will authenticate the identity of every vehicle that has sent a vote and also determine if the reputation of the vehicle is above a given threshold, as shown in Fig. 5-9. After a pre-determined timeout, the voting algorithm will count the total number of votes it receives. If the number of valid votes meets the minimum requirement (e.g., 2/3 of total votes), the RSU will broadcast this event to other RSUs or send an request for emergency responses to traffic controller when necessary.



```
def vote_for_consensus(timeout):
    vote_map = self.init_voteMap()
    num_vote = 0
    while timeout > 0:
        if self.packetQueue():
            num_vote = num_vote +1
            packet = self.packetQueue.pop()
            pubKey_origin = packet.get_pubKey_origin()  #. Get the public key of the sender vehicle
            signature_origin = packet.get_digitalSig_origin()  #. Get the digital signature for the message received

            #1. Check if the digital signature included in the message received is valid
            if self.check_Signature(signature_origin,pubKey_origin) == False:
                continue

            #2. Check if the reputation of the threshold is above a pre-defined threshould R_T
            if self.check_reputation(self.get_orgin_reputation(pubKey_origin,R_T)) == False:
                continue
            event = packet.getEvent()
            vote_map[event] = ote_map[event] + 1
        timeout = tiimeout - self.processtime()
    if self.max_vote(vote_map) > num_vote/3:
        event_maxvote = self.get_event_for_maxvote(vote_map)
        self.broadcast_event(event_maxvote)
    return
```

Figure 5-9: The voting algorithm for consensus formation based on Proof-of-Travel protocol

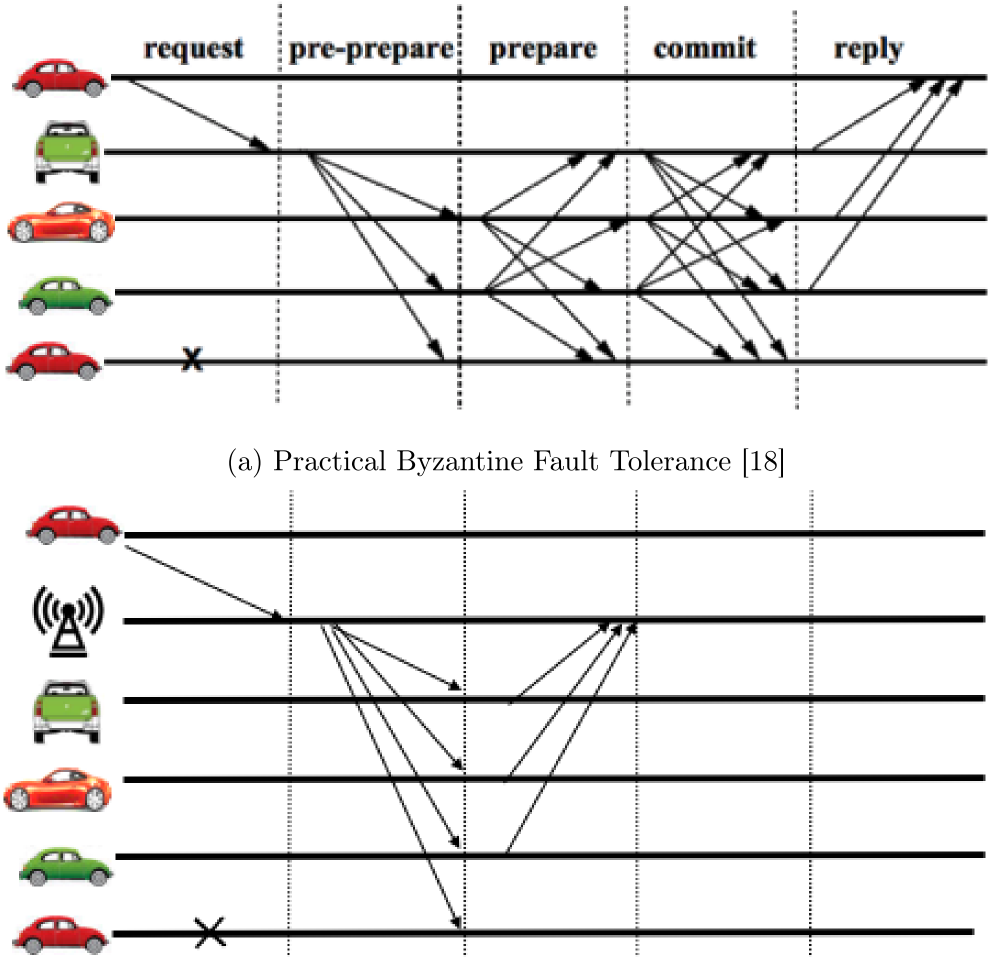
Scalability for POT vs. Other consensus algorithms

To evaluate the scalability of the POT protocol, we compare it with previous proof-based and voting-based protocols that are widely used. Table 5.2 compares POT and widely used proof-based protocols, such as among proof-of-work (POW) [82] and proof-of-stake (POS) [63].

Table 5.2: A comparison among different consensus protocols

Criteria	**POW**	**POS**	**POT**
Burdens for attackers	Computation power	Coin deposit	Spatial movement
Eligibility for voting	Computation power	Stake	Spatial movement

A comparison between the POT protocol and voting-based protocols, practical Byzantine Fault Tolerance (PBFT) in this case, is given in Fig. 5-10. In PBFT, each node has to communicate with all other nodes and wait for confirmations from each until a consensus is formed, which leads to a time complexity of $O(n^2)$. On the other hand, POT relies on a trusted entity (i.e., RSU) to collect all votes from vehicles and thus can reduce communication overhead.



(a) Practical Byzantine Fault Tolerance [18]

(b) Proof-of-Travel

Figure 5-10: A comparison between BPFT and POT regarding communication overheads

POT's role in mitigating adversaries

The effectiveness of POT's capability in mitigating intentionally tampering with traffic event reports can be understood by estimating the cost needed for a group of colluding adversaries to "win" in the consensus formation process. POT reduces the likelihood of adversarial behaviors by increasing the cost that an adversary group needs to take. As a result, rational adversaries will lose or at least have less economic incentives of initiating attacks.

The tradeoff between adversaries' winning probability and the total cost of winning shows how POT discourages attacks. Fig. 5-11 shows the tradeoff between the cost

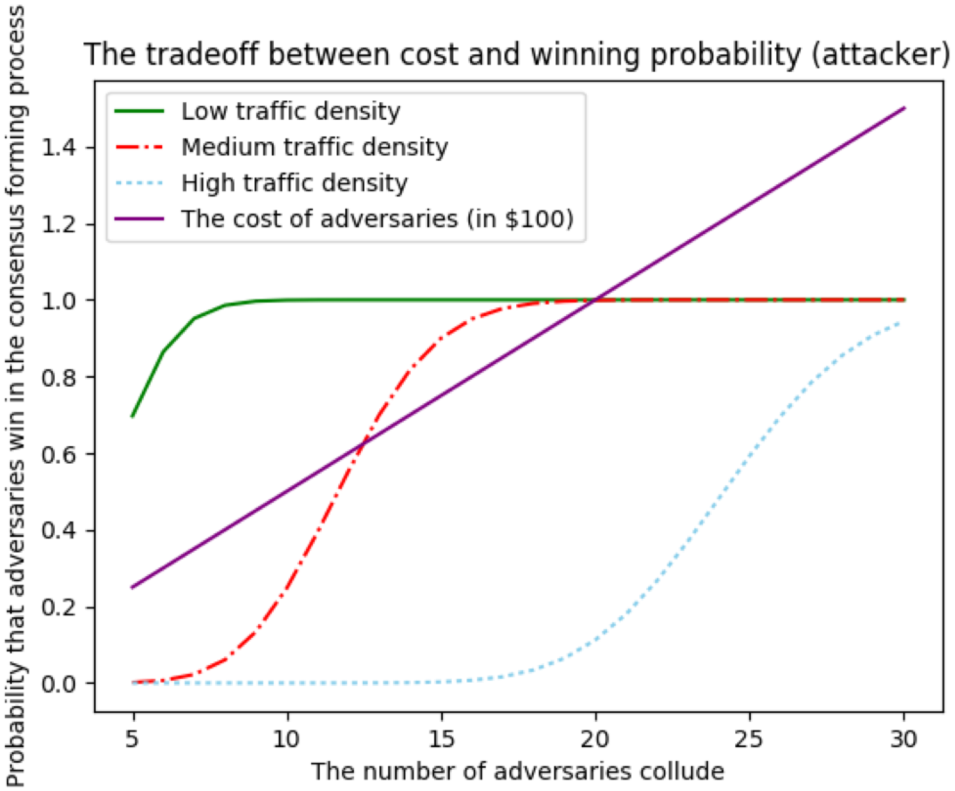


Figure 5-11: The tradeoff between the cost and probability of winning from adversaries' perspective. Note when POT is adopted to decide whether a vehicle is eligible to vote, the malicious group need to have more nodes and thus pay more to win in a the voting

an adversary group needs to take and the chance of winning under different traffic density conditions. The key idea is to leverage the unavoidable cost due to travel (i.e., burdens of spatial movement) that is incurred by the POT protocol for every vehicle. From an adversaries' perspective, a higher traffic density in a local region will reduce an adversary group's chance of winning or incur extra cost for the group compared to medium and low traffic conditions. In other words, POT is scalable to high traffic conditions as the market penetration of connected vehicles increases. To be more specific,

- Relationship between traffic density and adversaries' winning probability: The higher the traffic density, the more difficult it is for an adversary group to win. This is because the constraint on the number of vehicles who must vote for the same event due to the adoption of the voting rule that "majority wins" (in Fig. 5-8). In low traffic-density conditions, fewer malicious vehicles are required to vote for an adversary group to win. However, the size of the adversary group must be greater than 17 and 30 vehicles for medium and high traffic densities respectively.

- Relationship between the size of the adversary group and the cost or burdens for them to win: As the number of malicious vehicles participating in voting increases, the cost measured in U.S. dollars will also increase linearly. The detailed process of deriving the cost are discussed later; the increase in the overall cost puts the adversary group in a "game" situation as a large group size (i.e., more malicious vehicles participate in voting) can increase the probability to win in high-density scenarios, but the adversarial group as whole must pay more. Under the assumption that adversaries are rational, they must seek a balance between the cost and reward of being malicious. With sufficient cost, there will be no economic incentive to initiate such cyber attacks.

Under the assumption that multiple adversaries will collude in reporting malicious events in a local region, the adversaries' winning probability and adversarial cost under three traffic densities can be derived as follows:

The probability that an adversary group win is $Pr(n_r < 2 * n_a + 1)$, where n_r denotes the number of normal or benign vehicles participate in the voting and is represented as a random variable, n_a denotes the number of vehicles in the adversary group.

Proof. If the rule of "majority win" is adopted for consensus formation [64], for a colluding adversarial group to win, the number of malicious vehicles n_a must be greater than half of the number of normal vehicles n_r that participate in the voting process [18]. Assuming n_r is influenced by the density of traffic, then the probability that adversaries win is $Pr(n_r < 2 * n_a + 1)$. By further assuming that n_r follows $Poisson(\lambda)$, where λ denotes traffic density and thus the number of regular vehicles in a given local region at at short period of time (we ignore the number of vehicles leave during the same period of time), the probability that adversary win as the number of malicious vehicles increases can be generated, as shown in Fig. 5-11. A detailed derivation of the actual distribution of n_r is out of the scope of this thesis, but it does not change the constraints that the POT imposes on adversaries' behaviors as an adversary must physically move on the road in order to acquire verifiable mileages testified by RSUs. $\square$

The total cost that the adversary group needs to take to win can be calculated as $C_{adv} = \frac{R_T * C_m * n_a}{\alpha}$, where R_T is the threshold reputation score for being eligible to vote, C_m denotes the cost per mileage (vehicle's travel cost), and α is the reputation parameters for determining a vehicle's reputation based on its verifiable mileage traveled defined earlier.

Proof. Consider that each malicious node (denoted as v_i) must have a reputation score greater than a pre-determined threshold ($R_i > R_T$) to be eligible to vote based on the voting algorithm defined in Fig. 5-9. Therefore, we must have $\alpha * M_i > R_T$ for each malicious vehicle according to the stage-3 of the POT protocol. In other words, the total verifiable mileage that each malicious node traveled (M_i) must be at least $\frac{R_T}{\alpha}$. If we assume that the cost per mileage is C_m, then the cost for each individual malicious vehicle is $\frac{R_T * C_m}{\alpha}$. Therefore, the total cost for an adversary group with size n_a to win is $\frac{R_T * C_m * n_a}{\alpha}$. $\square$

The performance of POT-based voting

To evaluate the performance of voting algorithm based on the POT protocol (in Fig. 5-9), We implement the POT protocol and the voting algorithm in the V2X Simulation Runtime Infrastructure (VSimRTI) [105] and simulate the them in a two-lane highway with V2X connectivity.

Fig. 5-12 shows the performance of the scheme under different settings and traffic density conditions. In particular, we are interested in the latency of forming a consensus (i.e., the time duration between when an event occurs and when RSU has received enough valid votes to confirm the authenticity, correctness, or accuracy of the event)

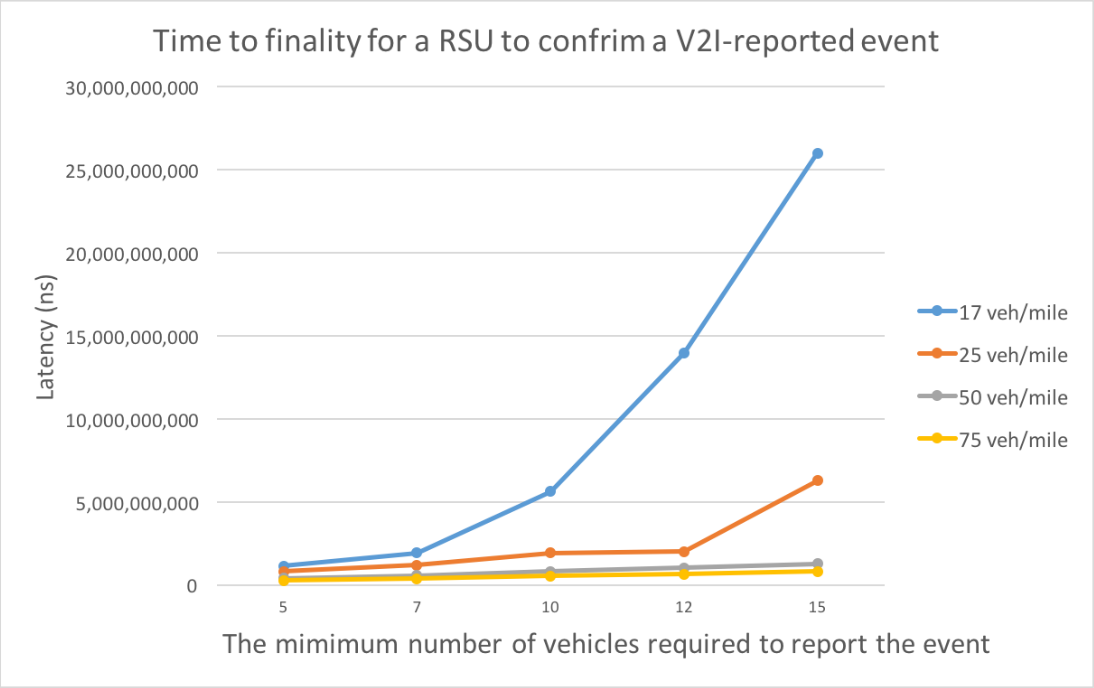


Figure 5-12: The latency of the POT-based consensus voting. This corresponds to the time it takes for the RSU and vehicles participating in voting to form the consensus on the authenticity and integrity of a V2I-reported traffic event. The RSU will report the event to the traffic management center for emergency responses if necessary.

when we vary the minimum number of votes required. We assume 50 percent of vehicles on the road support the POT protocol, and these vehicles have accumulated enough proofs from the previous path of movement when the event occurs.

The tradeoff between security and performance shown in Fig. 5-12 is crucial for using POT in any voting-based consensus. First, more vehicle votes required by the consensus algorithm (minimum number of votes required) means that the voting scheme installed on RSUs can tolerate more forgery V2I events reported by malicious vehicles. Second, under low traffic density conditions (e.g., 17 vehicles per miles) or when the V2X penetration rate is low (fewer vehicles support V2X and thus the POT protocol), it takes much more time to form a consensus if the RSU requires a higher number of minimum votes. For the requirement of 15 minimum number of votes, it takes almost 25 seconds for an RSU to confirm a high-criticality event when the density is equal to 17 vehicles per miles (Fig. 5-12), while it only takes around 5 seconds for the density of 50 miles per miles.

Chapter 6

Conclusions and Future Work

This thesis develops a security engineering framework that allows multiple engineering teams with OEMs to jointly identify cyber and physical threats in a systematic way. By developing two proof-of-concept designs, including on-board optical sensors by using light polarization properties and a Proof-of-Travel protocol for trust management in V2X communication, the thesis illustrate that design goals such as safety, security, and performance can be achieved in a more cost-effective ways. The thesis work can be extended in three aspects in the future.

- Implementing the proposed security engineering framework in OEMs or suppliers in the automotive domain. Experimentally evaluating the effectiveness of this approach in reducing time and manpower spent in threat mitigation.

- Incorporating the polarization-based object recognition technology into commercial products such as Lidar and camera. To achieve that, more real world testing are needed for collecting polarization signatures from different objects and materials.

- Explore the behavioral foundations of Proof of Travel. In particular, the assumption that a benign CAV (owned by customers) is willing to participate in reporting traffic events needed to be evaluated by using tools from economic and social perspectives such as game theory. On the other hand, it is worth exploring gamification mechanisms for the adoption of connected and automated vehicles in order to change customers' perception on the benefits of using CAVs. The Proof-of-Travel protocol developed in this thesis lays out the technical foundations for ensuring data for customer behaviors in using CAVs can be recorded and shared in a secured manner.

Bibliography

[1] Quartz Africa. Uber drivers in lagos are using a fake gps app to inflate rider fares, 2017.

[2] National Security Agency. Defense in depth:a practical strategy for achieving information assurance in today's highly networked environments, 2010.

[3] Muhammad Arshad, Zahid Ullah, Naveed Ahmad, Muhammad Khalid, Haithiam Criuckshank, and Yue Cao. A survey of local/cooperative-based malicious information detection techniques in vanets. *EURASIP Journal on Wireless Communications and Networking*, 2018(1):62, 2018.

[4] Audi. Audi connect, 2016.

[5] Audi. Experience audi: Autonomous driving, 2019.

[6] Tampa Hillsborough Expressway Authority. Thea connected vehicle pilot, 2019.

[7] Rajesh P Barnwal and Soumya K Ghosh. Heartbeat message based misbehavior detection scheme for vehicular ad-hoc networks. In *2012 International Conference on Connected Vehicles and Expo (ICCVE)*, pages 29–34. IEEE, 2012.

[8] Mohamed Baza, Mahmoud Nabil, Mohamed Mohamed Elsalih Abdelsalam Mahmoud, Niclas Bewermeier, Kemal Fidan, Waleed Alasmary, and Mohamed Abdallah. Detecting sybil attacks using proofs of work and location in vanets. *IEEE Transactions on Dependable and Secure Computing*, 2020.

[9] Massimo Bertozzi, Alberto Broggi, and Alessandra Fascioli. Vision-based intelligent vehicles: State of the art and perspectives. *Robotics and Autonomous systems*, 32(1):1–16, 2000.

[10] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

[11] Norbert Bißmeyer. *Misbehavior detection and attacker identification in vehicular ad-hoc networks.* PhD thesis, Technische Universität, 2014.

[12] Norbert Bißmeyer, Joël Njeukam, Jonathan Petit, and Kpatcha M Bayarou. Central misbehavior evaluation for vanets based on mobility data plausibility. In *Proceedings of the ninth ACM international workshop on Vehicular inter-networking, systems, and applications*, pages 73–82. ACM, 2012.

[13] Subir Biswas and Jelena Mišić. Location-based anonymous authentication for vehicular communications. In *2011 IEEE 22nd International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1213–1217. IEEE, 2011.

[14] Rachel Blin, Samia Ainouz, Stéphane Canu, and Fabrice Meriaudeau. Road scenes analysis in adverse weather conditions by polarization-encoded images and adapted deep learning. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 27–32. IEEE, 2019.

[15] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z Morley Mao. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 2267–2281, 2019.

[16] Zhen Cao, Jiejun Kong, Uichin Lee, Mario Gerla, and Zhong Chen. Proof-of-relevance: Filtering false data via authentic consensus in vehicle ad-hoc networks. In *IEEE INFOCOM Workshops 2008*, pages 1–6. IEEE, 2008.

[17] Srdjan Capkun and Jean-Pierre Hubaux. Secure positioning of wireless devices with application to sensor networks. In *IEEE infocom*, number CONF, 2005.

[18] Miguel Castro, Barbara Liskov, et al. Practical byzantine fault tolerance. In *OSDI*, volume 99, pages 173–186, 1999.

[19] Kentucky Transportation Center. Traffic control procedures for emergency responders, 2006.

[20] Shan Chang, Yong Qi, Hongzi Zhu, Jizhong Zhao, and Xuemin Shen. Footprint: detecting sybil attacks in urban vehicular networks. *IEEE Transactions on Parallel and Distributed Systems*, 23(6):1103–1114, 2011.

[21] Anupam Chattopadhyay and Kwok-Yan Lam. Autonomous vehicle: Security by design. *arXiv preprint arXiv:1810.00545*, 2018.

[22] David Chaum and Eugène Van Heyst. Group signatures. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 257–265. Springer, 1991.

[23] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, Tadayoshi Kohno, et al. Comprehensive experimental analyses of automotive attack surfaces. In *USENIX Security Symposium*, volume 4, pages 447–462. San Francisco, 2011.

[24] Chen Chen, Xin Wang, Weili Han, and Binyu Zang. A robust detection of the sybil attack in urban vanets. In *2009 29th IEEE International Conference on Distributed Computing Systems Workshops*, pages 270–276. IEEE, 2009.

[25] Qi Alfred Chen, Yucheng Yin, Yiheng Feng, Z Morley Mao, and Henry X Liu. Exposing congestion attack on emerging connected vehicle based traffic signal control. In *NDSS*, 2018.

[26] Cornell SL Chun and Firooz A Sadjadi. Polarimetric laser radar target classification. *Optics letters*, 30(14):1806–1808, 2005.

[27] Luis Cintron, Scott Graham, Douglas Hodson, and Barry Mullins. Distributed-ledger based event attestation for intelligent transportation systems. In *International Conference on Cyber Warfare and Security*, pages 565–XI. Academic Conferences International Limited, 2019.

[28] SAE On-Road Automated Vehicle Standards Committee et al. Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems. *SAE Standard J*, 3016:1–16, 2014.

[29] Ford Motor Company. Ford, walmart and postmates team up for self-driving goods delivery, 2018.

[30] Ford Motor Company. A matter of trust: Ford's approach to developing self-driving vehicles, 2018.

[31] Jin Cui, Lin Shen Liew, Giedre Sabaliauskaite, and Fengjun Zhou. A review on safety failures, security attacks, and available countermeasures for autonomous vehicles. *Ad Hoc Networks*, 90:101823, 2019.

[32] Daimler. 2019 safety first for automated driving, 2019.

[33] Stefano De Angelis, Leonardo Aniello, Roberto Baldoni, Federico Lombardi, Andrea Margheri, and Vladimiro Sassone. Pbft vs proof-of-authority: Applying the cap theorem to permissioned blockchain. 2018.

[34] Mahmoud Hashem Eiza and Qiang Ni. Driving with sharks: Rethinking connected vehicles with vehicle cybersecurity. *IEEE Vehicular Technology Magazine*, 12(2):45–51, 2017.

[35] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

[36] Nathaniel Fairfield, Joshua Seth Herbach, and Vadim Furman. Remote assistance for autonomous vehicles in predetermined situations, August 1 2017. US Patent 9,720,410.

[37] Wang Fan, Samia Ainouz, Fabrice Meriaudeau, and Abdelaziz Bensrhair. Polarization-based car detection. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3069–3073. IEEE, 2018.

[38] Xia Feng, Chun-yan Li, De-xin Chen, and Jin Tang. A method for defensing against multi-source sybil attacks in vanet. *Peer-to-Peer Networking and Applications*, 10(2):305–314, 2017.

[39] Yiheng Feng, Shihong Huang, Qi Alfred Chen, Henry X Liu, and Z Morley Mao. Vulnerability of traffic control system under cyber-attacks using falsified data. In *97th Annual Meeting of the Transportation Research Board*, 2018.

[40] Matthias Gerlach. Trust for vehicular applications. In *Eighth International Symposium on Autonomous Decentralized Systems (ISADS'07)*, pages 295–304. IEEE, 2007.

[41] Branden Ghena, William Beyer, Allen Hillaker, Jonathan Pevarnek, and J Alex Halderman. Green lights forever: Analyzing the security of traffic infrastructure. In *8th {USENIX} Workshop on Offensive Technologies ({WOOT} 14)*, 2014.

[42] Kannan Govindan and Prasant Mohapatra. Trust computations and trust dynamics in mobile adhoc networks: A survey. *IEEE Communications Surveys & Tutorials*, 14(2):279–298, 2011.

[43] Object Management Group. Unified modeling language, 2017.

[44] Object Management Group. Omg system modeling language, 2019.

[45] Hamssa Hasrouny, Abed Ellatif Samhat, Carole Bassil, and Anis Laouiti. Vanet security challenges and solutions: A survey. *Vehicular Communications*, 7:7–20, 2017.

[46] Eugene Hecht. *Optik*. Walter de Gruyter GmbH & Co KG, 2018.

[47] Martin J How and N Justin Marshall. Polarization distance: a framework for modelling object detection by polarization vision systems. *Proceedings of the Royal Society B: Biological Sciences*, 281(1776):20131632, 2014.

[48] Intel. Developing amazing android automotive in-vehicle infotainment experiences, 2018.

[49] Shahrear Iqbal, Anwar Haque, and Mohammad Zulkernine. Towards a security architecture for protecting connected vehicles from malware. In *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, pages 1–5. IEEE, 2019.

[50] Rob Millerb Ishtiaq Roufa, Hossen Mustafaa, Sangho Ohb Travis Taylora, Wenyuan Xua, Marco Gruteserb, Wade Trappeb, and Ivan Seskarb. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study. In *19th USENIX Security Symposium, Washington DC*, pages 11–13, 2010.

[51] ISO ISO. Pas 21448-road vehicles-safety of the intended functionality. *International Organization for Standardization*, 2019.

[52] ISO ISO and CD SAE. 21434-road vehicles—cybersecurity engineering. *ISO: London, UK*, 2019.

[53] ISO26262 ISO. 26262: Road vehicles-functional safety. *International Standard ISO/FDIS*, 26262, 2011.

[54] Ahmad Jalali and Mohammad Ali Hadavi. Software security analysis based on the principle of defense-in-depth. In *2018 15th International ISC (Iranian Society of Cryptology) Conference on Information Security and Cryptology (ISCISC)*, pages 1–6. IEEE, 2018.

[55] Audun Jøsang, Ross Hayward, and Simon Pope. Trust network analysis with subjective logic. In *Proceedings of the 29th Australasian Computer Science Conference-Volume 48*, pages 85–94. Australian Computer Society, Inc., 2006.

[56] John B Kenney. Dedicated short-range communications (dsrc) standards in the united states. *Proceedings of the IEEE*, 99(7):1162–1182, 2011.

[57] Faiq Khalid, Muhammad Abdullah Hanif, Semeen Rehman, Rehan Ahmed, and Muhammad Shafique. Trisec: training data-unaware imperceptible security attacks on deep neural networks. In *2019 IEEE 25th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, pages 188–193. IEEE, 2019.

[58] Faiq Khalid, Muhammad Abdullah Hanif, Semeen Rehman, and Muhammad Shafique. Security for machine learning-based systems: Attacks and challenges during training and inference. In *2018 International Conference on Frontiers of Information Technology (FIT)*, pages 327–332. IEEE, 2018.

[59] Irfan Khan, Gia-Minh Hoang, and Jérôme Härri. Rethinking cooperative awareness for future v2x safety-critical applications. In *2017 IEEE Vehicular Networking Conference (VNC)*, pages 73–76. IEEE, 2017.

[60] Dara Khosrowshahi. 2016 data security incident, 2016.

[61] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, et al. Experimental security analysis of a modern automobile. In *2010 IEEE Symposium on Security and Privacy*, pages 447–462. IEEE, 2010.

[62] Ramakrishna Kotla, Lorenzo Alvisi, Mike Dahlin, Allen Clement, and Edmund Wong. Zyzzyva: speculative byzantine fault tolerance. *ACM SIGOPS Operating Systems Review*, 41(6):45–58, 2007.

[63] Jae Kwon. Tendermint: Consensus without mining. *Draft v. 0.6, fall*, 1(11), 2014.

[64] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. In *Concurrency: the Works of Leslie Lamport*, pages 203–226. 2019.

[65] Tim Leinmüller, Christian Maihöfer, Elmar Schoch, and Frank Kargl. Improved security in geographic ad hoc routing through autonomous position verification. In *Proceedings of the 3rd international workshop on Vehicular ad hoc networks*, pages 57–66. ACM, 2006.

[66] Tim Leinmuller, Elmar Schoch, and Frank Kargl. Position verification approaches for vehicular ad hoc networks. *IEEE Wireless Communications*, 13(5):16–21, 2006.

[67] Nancy Leveson. *Engineering a safer world: Systems thinking applied to safety.* MIT press, 2011.

[68] Jesse Sol Levinson, Timothy David Kentley, Gabriel Thurston Sibley, Rachad Youssef Gamara, Ashutosh Gajanan Rege, and Gary Linscott. Teleoperation system and method for trajectory modification of autonomous vehicles, November 29 2016. US Patent 9,507,346.

[69] Václav Linkov, Petr Zámečník, Darina Havlíčková, and Chih-Wei Pai. Human factors in the cybersecurity of autonomous cars: trends in current research. *Frontiers in psychology*, 10:995, 2019.

[70] Huiye Liu, Chung-Wei Lin, Eunsuk Kang, Shinichi Shiraishi, and Douglas M Blough. A byzantine-tolerant distributed consensus algorithm for connected vehicles using proof-of-eligibility. In *Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 225–234, 2019.

[71] Jiajia Liu, Shubin Zhang, Wen Sun, and Yongpeng Shi. In-vehicle network attacks and countermeasures: Challenges and future directions. *IEEE Network*, 31(5):50–58, 2017.

[72] Tong Liu, Zhongchao Shi, Cheng Zhong, Yuan Liu, and Yi Chen. Method and system for detecting vehicle position by employing polarization image, March 17 2015. US Patent 8,983,126.

[73] David Tse-Zhou Lu, Calvin Karl Johnson, and Renaud-Roland Hubert. Unlock and authentication for autonomous vehicles, November 24 2015. US Patent 9,194,168.

[74] Rongxing Lu, Xiaodong Lin, Xiaohui Liang, and Xuemin Shen. A dynamic privacy-preserving key management scheme for location-based services in vanets. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):127–139, 2011.

[75] Francesco Malandrino, Carlo Borgiattino, Claudio Casetti, Carla-Fabiana Chiasserini, Marco Fiore, and Roberto Sadao. Verification and inference of positions in vehicular networks through anonymous beaconing. *IEEE transactions on mobile computing*, 13(10):2415–2428, 2014.

[76] Nisha Malik, Priyadarsi Nanda, Xiangjian He, and RenPing Liu. Trust and reputation in vehicular networks: A smart contract-based approach. In *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 34–41. IEEE, 2019.

[77] McAfee. Model hacking adas to pave safer roads for autonomous vehicles, 2020.

[78] Microsoft. The stride threat model, 2009.

[79] Microsoft. Microsoft security development lifecycle, 2019.

[80] Charlie Miller. Lessons learned from hacking a car. *IEEE Design & Test*, 36(6):7–9, 2019.

[81] Umar Farooq Minhas, Jie Zhang, Thomas Tran, and Robin Cohen. Intelligent agents in mobile vehicular ad-hoc networks: Leveraging trust modeling based on direct experience with incentives for honesty. In *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, volume 2, pages 243–247. IEEE, 2010.

[82] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.

[83] Ben Nassi, Dudi Nassi, Raz Ben-Netanel, Yisroel Mirsky, Oleg Drokin, and Yuval Elovici. Phantom of the adas: Phantom attacks on driver-assistance systems, 2020.

[84] Eric Newcomer. Uber paid hackers to delete stolen data on 57 million people, 2017.

[85] NTSB. Preliminary report, highway, hwy18mh010. *National Transportation Safety Board*, 2018.

[86] NTSB. Collision between car operating with partial driving automation and truck-tractor semitrailer delray beach, florida. *National Transportation Safety Board*, 2019.

[87] Marcus Obst, Laurens Hobert, and Pierre Reisdorf. Multi-sensor data fusion for checking plausibility of v2v communications by vision-based multiple-object tracking. In *2014 IEEE Vehicular Networking Conference (VNC)*, pages 143–150. IEEE, 2014.

[88] Department of Transportation. *Connected Vehicle Pilot Deployment Program*, 2019 (accessed April 15, 2019).

[89] U.S. Department of Transportation. Preparing for the future of transportation: Automated vehicles 3.0, 2018.

[90] U.S. Department of Transportation. Real-world deployment of connected vehicles: Challenges and lessons learned, 2019.

[91] U.S. Department of Transportation. Traffic incident management, 2020.

[92] Aleksandr Ometov and Sergey Bezzateev. Multi-factor authentication: A survey and challenges in v2x applications. In *2017 9th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pages 129–136. IEEE, 2017.

[93] OnStar. Onstar connected services, 2016.

[94] Soyoung Park, Baber Aslam, Damla Turgut, and Cliff C Zou. Defense against sybil attack in the initial deployment stage of vehicular ad hoc network based on roadside unit support. *Security and Communication Networks*, 6(4):523–538, 2013.

[95] Youngho Park, Kyung-Hyune Rhee, and Chul Sur. A secure and location assurance protocol for location-aware services in vanets. In *2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 456–461. IEEE, 2011.

[96] Simon Parkinson, Paul Ward, Kyle Wilson, and Jonathan Miller. Cyber threats facing autonomous and connected vehicles: Future challenges. *IEEE transactions on intelligent transportation systems*, 18(11):2898–2915, 2017.

[97] Michael C Pasqual and Kerri L Cahoy. Active polarimetric measurements for identification and characterization of space debris. *IEEE Transactions on Aerospace and Electronic Systems*, 53(6):2706–2717, 2017.

[98] Jonathan Petit. Automated vehicles cybersecurity: Summary avs'17 and stakeholder analysis. In *Road Vehicle Automation 5*, pages 171–181. Springer, 2019.

[99] Jonathan Petit, Michael Feiri, and Frank Kargl. Revisiting attacker model for smart vehicles. In *2014 IEEE 6th International Symposium on Wireless Vehicular Communications (WiVeC 2014)*, pages 1–5. IEEE, 2014.

[100] Jonathan Petit, Florian Schaub, Michael Feiri, and Frank Kargl. Pseudonym schemes in vehicular networks: A survey. *IEEE communications surveys & tutorials*, 17(1):228–255, 2014.

[101] Jonathan Petit and Steven E Shladover. Potential cyberattacks on automated vehicles. *IEEE Transactions on Intelligent transportation systems*, 16(2):546–556, 2014.

[102] Jonathan Petit, Bas Stottelaar, Michael Feiri, and Frank Kargl. Remote attacks on automated vehicles sensors: Experiments on camera and lidar. *Black Hat Europe*, 11:2015, 2015.

[103] Jonathan Petit, Bas Stottelaar, Michael Feiri, and Frank Kargl. Self-driving and connected cars: Fooling sensors and tracking drivers. *Black Hat Europe*, 2015.

[104] Kyle Post and Christopher K Davey. Integrating sotif and agile systems engineering. Technical report, SAE Technical Paper, 2019.

[105] Robert Protzmann, Björn Schünemann, and Ilja Radusch. Simulation of convergent networks for intelligent transport systems with vsimrti. *Networking Simulation for Intelligent Transportation Systems: High Mobile Wireless Nodes*, pages 1–28, 2017.

[106] Adnan Qayyum, Muhammad Usama, Junaid Qadir, and Ala Al-Fuqaha. Securing connected & autonomous vehicles: Challenges posed by adversarial machine learning and the way forward. *arXiv preprint arXiv:1905.12762*, 2019.

[107] Yvain Quéau, Florian Leporcq, Alexis Lechervy, and Ayman Alfalou. Learning to classify materials using mueller imaging polarimetry. In *Fourteenth International Conference on Quality Control by Artificial Vision*, volume 11172, page 111720Z. International Society for Optics and Photonics, 2019.

[108] Jens Rasmussen. Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models. *IEEE transactions on systems, man, and cybernetics*, (3):257–266, 1983.

[109] Kui Ren, Qian Wang, Cong Wang, Zhan Qin, and Xiaodong Lin. The security of autonomous driving: Threats, defenses, and future directions. *Proceedings of the IEEE*, 2019.

[110] ridester. Fake gps: Uber's deactivation rampage, 2017.

[111] Karim Rostamzadeh, Hasen Nicanfar, Narjes Torabi, Sathish Gopalakrishnan, and Victor CM Leung. A context-aware trust-based information dissemination framework for vehicular networks. *IEEE Internet of Things journal*, 2(2):121–132, 2015.

[112] Noah A Rubin, Gabriele D'Aversa, Paul Chevalier, Zhujun Shi, Wei Ting Chen, and Federico Capasso. Matrix fourier optics enables a compact full-stokes polarization camera. *Science*, 365(6448):eaax1839, 2019.

[113] Sushmita Ruj, Marcos A Cavenaghi, Zhen Huang, Amiya Nayak, and Ivan Stojmenovic. On data-centric misbehavior detection in vanets. In *2011 IEEE Vehicular Technology Conference (VTC Fall)*, pages 1–5. IEEE, 2011.

[114] Jordi Sabater and Carles Sierra. Regret: reputation in gregarious societies. In *Agents*, volume 1, pages 194–195, 2001.

[115] Firooz Sadjadi and Farzad Sadjadi. Passive polarimetric information processing for target classification. In *Augmented Vision Perception in Infrared*, pages 37–61. Springer, 2009.

[116] J SAE. 3061: Cybersecurity guidebook for cyber-physical vehicle systems. 2016. *Society for automotive engineers*, 2016.

[117] Fatih Sakiz and Sevil Sen. A survey of attacks and detection mechanisms on intelligent transportation systems: Vanets and iov. *Ad Hoc Networks*, 61:33–50, 2017.

[118] Spyridon Samonas and David Coss. The cia strikes back: Redefining confidentiality, integrity and availability in security. *Journal of Information System Security*, 10(3), 2014.

[119] Beth Schaefer, Edward Collett, Robert Smyth, Daniel Barrett, and Beth Fraher. Measuring the stokes polarization parameters. *American Journal of Physics*, 75(2):163–168, 2007.

[120] Robert K Schmidt, Tim Leinmüller, Elmar Schoch, Albert Held, and Günter Schäfer. Vehicle behavior analysis to enhance security in vanets. In *Proceedings of the 4th IEEE Vehicle-to-Vehicle Communications Workshop (V2VCOM2008)*, 2008.

[121] Bruce Schneier. Attack trees. *Dr. Dobb's journal*, 24(12):21–29, 1999.

[122] Barry Sheehan, Finbarr Murphy, Martin Mullins, and Cian Ryan. Connected and autonomous vehicles: A cyber-risk classification framework. *Transportation research part A: policy and practice*, 124:523–536, 2019.

[123] Hocheol Shin, Dohyun Kim, Yujin Kwon, and Yongdae Kim. Illusion and dazzle: Adversarial optical channel exploits against lidars for automotive applications. In *International Conference on Cryptographic Hardware and Embedded Systems*, pages 445–467. Springer, 2017.

[124] Ankit Singh and Hervais C Simo Fhom. Restricted usage of anonymous credentials in vehicular ad hoc networks for misbehavior detection. *International Journal of Information Security*, 16(2):195–211, 2017.

[125] Steven So, Prinkle Sharma, and Jonathan Petit. Integrating plausibility checks and machine learning for misbehavior detection in vanet. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 564–571. IEEE, 2018.

[126] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks. In *Advances in neural information processing systems*, pages 3517–3529, 2017.

[127] George Gabriel Stokes. On the composition and resolution of streams of polarized light from different sources. *Transactions of the Cambridge Philosophical Society*, 9:399, 1851.

[128] Bas GB Stottelaar. Practical cyber-attacks on autonomous vehicles. Master's thesis, University of Twente, 2015.

[129] Dajiang Suo and Sanjay E Sarma. Real-time trust-building schemes for mitigating malicious behaviors in connected and automated vehicles. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1142–1149. IEEE, 2019.

[130] Dajiang Suo and Sanjay E Sarma. A multi-pola active polarimetry for object recognition for robotics applications. *working paper in progress*, 2020.

[131] Dajiang Suo, Josh Siegel, and Alexander Soley. Driving data dissemination: The "terms"governing connected car information (submitted). *IEEE Intelligent Transportation Systems Magazine*, 2020.

[132] Dajiang Suo, Joshua E Siegel, and Sanjay E Sarma. Merging safety and cybersecurity analysis in product design. *IET Intelligent Transport Systems*, 12(9):1103–1109, 2018.

[133] Dajiang Suo, Sarra Yako, Mathew Boesch, and Kyle Post. Integrating stpa into iso 26262 process for requirement development. Technical report, SAE Technical Paper, 2017.

[134] Junko Takahashi. An overview of cyber security for connected vehicles. *IEICE TRANSACTIONS on Information and Systems*, 101(11):2561–2575, 2018.

[135] Tampa. Connected vehicle pilot deployment program phase i comprehensive pilot deployment plan, 2016.

[136] The Tesla Team. A tragic loss. *Uber*, 2016.

[137] Tencent. Experimental security research of tesla autopilot, 2019.

[138] Tesla. Future of driving: Advanced sensor coverage, 2019.

[139] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.

[140] TMZ. Pissed off l.a. homeowners waze is the devil, 2014.

[141] Daniel Tokody, Attila Albini, Laszlo Ady, Zoltan Rajnai, and Ferenc Pongracz. Safety and security through the design of autonomous intelligent vehicle systems and intelligent infrastructure in the smart city. *Interdisciplinary Description of Complex Systems: INDECS*, 16(3-A):384–396, 2018.

[142] Rens W van der Heijden, Thomas Lukaseder, and Frank Kargl. Veremi: A dataset for comparable evaluation of misbehavior detection in vanets. In *International Conference on Security and Privacy in Communication Systems*, pages 318–337. Springer, 2018.

[143] Rens Wouter Van der Heijden. *Misbehavior detection in cooperative intelligent transport systems.* PhD thesis, Universität Ulm, 2018.

[144] Rens Wouter van der Heijden, Stefan Dietzel, Tim Leinmüller, and Frank Kargl. Survey on misbehavior detection in cooperative intelligent transportation systems. *IEEE Communications Surveys & Tutorials*, 21(1):779–811, 2018.

[145] Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 2017.

[146] Lawrence B. Wolff. Polarization-based material classification from specular reflection. *IEEE transactions on pattern analysis and machine intelligence*, 12(11):1059–1071, 1990.

[147] Wai Wong, Shihong Huang, Yiheng Feng, Qi Alfred Chen, Z Morley Mao, and Henry X Liu. Trajectory-based hierarchical defense model to detect cyber-attacks on transportation infrastructure. Technical report, 2019.

[148] Samuel Woo, Hyo Jin Jo, and Dong Hoon Lee. A practical wireless attack on the connected car and security protocol for in-vehicle can. *IEEE Transactions on intelligent transportation systems*, 16(2):993–1006, 2014.

[149] Wenyuan Xu, Chen Yan, Weibin Jia, Xiaoyu Ji, and Jianhao Liu. Analyzing and enhancing the security of ultrasonic sensors for autonomous vehicles. *IEEE Internet of Things Journal*, 5(6):5015–5029, 2018.

[150] Chen Yan, Wenyuan Xu, and Jianhao Liu. Can you trust autonomous vehicles: Contactless attacks against sensors of self-driving vehicle. *DEF CON*, 24, 2016.

[151] Yao-Tsung Yang, Li-Der Chou, Chia-Wei Tseng, Fan-Hsun Tseng, and Chien-Chang Liu. Blockchain-based traffic event validation and trust verification for vanets. *IEEE Access*, 7:30868–30877, 2019.

[152] Enoch Yeh, Junil Choi, Nuria G Prelcic, Chandra R Bhat, Robert W Heath, et al. Cybersecurity challenges and pathways in the context of connected vehicle systems. Technical report, University of Texas at Austin. Data-Supported Transportation Operations . . . , 2018.

[153] William Young and Nancy G Leveson. An integrated approach to safety and security based on systems theory. *Communications of the ACM*, 57(2):31–35, 2014.

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

[155] Giorgos Zacharia and Pattie Maes. Trust management through reputation mechanisms. *Applied Artificial Intelligence*, 14(9):881–907, 2000.

[156] Jithin Zacharias and Sibylle Fröschle. Misbehavior detection system in vanets using local traffic density. In *2018 IEEE Vehicular Networking Conference (VNC)*, pages 1–4. IEEE, 2018.

[157] Jie Zhang. A survey on trust management for vanets. In *2011 IEEE International Conference on Advanced Information Networking and Applications*, pages 105–112. IEEE, 2011.

[158] Tao Zhang and Luca Delgrossi. *Vehicle safety communications: protocols, security, and privacy*, volume 103. John Wiley & Sons, 2012.