博　士　論　文

Let the Truth Tell: Zero-Knowledge Proof Mechanisms to Realize Fact-Based Cooperative ITS

(事実に基づいた協調型ITSを実現するゼロ知識証明方法)



東京大学大学院
情報理工学系研究科
創造情報学専攻

48-167603　陶　冶

指導教員　江崎　浩　教授

令和 5 年　12 月

© Copyright by Ye Tao 2023.
All rights reserved.

Abstract

The Cooperative Intelligent Transportation System (C-ITS) represents a communication-based technological framework designed to facilitate the exchange of information among vehicles. This exchange encompasses details pertaining to their presence, observations, and intentions, thereby contributing to the realization of shared objectives, primarily safety and efficiency. Functioning as a distributed and ad-hoc system, C-ITS relies on both direct vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication protocols.

Given its dynamic topology and intermittent global connectivity, C-ITS is susceptible to various cybersecurity threats. Threats such as data manipulation, the creation of ghost vehicles, and the abuse of pseudonymous identities pose significant challenges. These threats can potentially lead to erroneous decision-making by the affected entities.

Misbehavior detection (MBD) is an endeavor to detect and mitigate cyber-attacks in C-ITS. MBD is categorized into two types: data-centric approaches leverage plausibility and consistency to identify potential cyber threats; node-based approaches rely on the past behavior and reputation of a vehicle to assess its likelihood of engaging in dishonest activities. However, both approaches exhibit limitations. Data-centric approaches are constrained to detecting predefined misbehaviors and are susceptible to false positives and negatives. On the other hand, node-centric approaches suffer from high latency in establishing and updating trust assessments, including blocking ongoing attacks.

In contrast, emerging technologies such as blockchain and cryptocurrencies establish consensus through verifiable facts and employ game theory to discourage dishonest behavior. These fact-based approaches differ markedly from traditional C-ITS methods, which often assume a prevailing honesty among individuals. Demonstrating efficacy across various industries, including healthcare, education, logistics, and government, fact-based approaches exhibit notable potential for enhancing C-ITS. Several works proposed in this dissertation leverage these fact-based methodologies.

In chapter 3, we introduced Flowsim, a novel simulator tailored for evaluating Connected and Autonomous Vehicles (CAV) behavior and data flow in large-scale scenarios. The primary objective of Flowsim is to expedite the development and assessment of cybersecurity measures and protocols for CAVs, ensuring their safe and secure deployment. Beyond its focus on CAV evaluation, Flowsim demonstrates versatility in various use cases, including protocol evaluation, cybersecurity assessment, traffic optimization, and policy development. The simulator's extensible and playable open environment encourages the exploration of innovative algorithms and strategies for CAVs, fostering collaboration and innovation in research communities. Flowsim shows its significance in addressing the limitations of existing simulators for CAV behavior evaluation, specifically for its modularity, extensibility, and performance.

In chapter 4, we introduced a novel deterministic cross-verification method named zero-knowledge Proof of Traffic (zk-PoT) within the context of cooperative perception. The zk-PoT protocol utilizes zero-knowledge proofs (ZKPs) to enable vehicles to independently prove their observations' existence, allowing remote parties to cross-verify these observations without relying on the ground truth. The zero-knowledge property ensures that no additional information is disclosed during this process, thereby safeguarding the location privacy of observed vehicles. Integration into existing cooperative perception standards, such as ISO and ETSI, requires minimal architectural changes and ensures backward compatibility. The study concludes with quantitative simulation analyses based on Flowsim in the scenario of a full-scaled Luxemburg city, demonstrating zk-PoT's high success rate, low decision latency, and moderate resource assumption. Threat analyses indicate resilience against common attacks, making it a valuable standalone method or complementary to existing standards and trust management models.

In chapter 5, we addressed the challenge of a type of Sybil attacks in C-ITS, caused by simultaneous usage of pseudonyms. We introduced a novel cryptographic protocol called zero-knowledge Proof of Distinct Identity (zk-PoDI). Based on the "locality assumption," wherein a vehicle need only prove its distinction from nearby vehicles rather than every vehicle globally, our proposed protocol zk-PoDI ensures that only a vehicle itself can prove it is not the owner of a given pseudonym locally, preventing Sybil attacks while preserving pseudonym unlinkability. The protocol's practicality is highlighted by its zero latency, moderate computation overhead, and negligible communication costs. zk-PoDI operates independently, without reliance on specific pseudonym system designs or additional knowledge. This feature together with the other practicality makes zk-PoDI ready to be integrated into C-ITS systems with any possible pseudonym designs.

Looking ahead, the exploration of fact-based approaches in C-ITS is poised to continue. Empowered by innovations like zero-knowledge proofs and other cutting-edge techniques, we can anticipate the resolution of numerous challenges, the validation of truth, and the safeguarding of privacy within this domain.

Acknowledgements

I am deeply grateful to my supervisor, Prof. Hiroshi Esaki, for his invaluable guidance and inspiration. He taught me that research should be driven by the needs of the society, not by the limits of our knowledge. With this vision, I ventured into the new field of Cooperative ITS, where I learned about intelligent transportation, vehicular networks, and autonomous driving, and tried to address the most pressing challenges.

I also owe a lot to my mentor, Assoc. Prof. Manabu Tsukada, who has been with me for almost 10 years. He is not only a brilliant researcher, but also a caring teacher and a good friend. He established his own laboratory from scratch, and nurtured it with his dedication and passion. He also helped me grow from a novice master student to a competent researcher, who can independently conduct research from problem identification to solution verification. He always gave me objective and constructive feedback, encouraged me to pursue my own research interests, and supported me in overcoming difficulties. He also shared with me his positive and optimistic attitude, and his rich experience in life and research. I will always cherish the memories of working with him, especially the time we spent in France.

I would also like to express my sincere appreciation to Dr. Ehsan Javanmardi, for his generous and patient help. He listened attentively to my half-baked ideas, quickly grasped their essence, and helped me clarify their logic. He also demanded high standards and integrity in my work, and guided me in writing and revising papers.

I am very thankful to all the committee members for their insightful comments and suggestions. I thank Prof. Kei Okada for his advice on the overall structure of the paper, Prof. Shigeru Chiba for his accurate understanding and feedback on my various work, Assoc. Prof. Shinpei Kato for his recommendation on incorporating the ITS market developments, and Assoc. Prof Masaki Ito for his tips on precise expression.

I would also like to thank all the members of the Tsukada lab, for creating a stimulating and enjoyable environment. I learned a lot from the meetings and discussions with them. I thank Dr. Jin Nakazato for his various support

in my research and paper writing, and Dr. Muhammad Asad for his assistance and help on my paper writing. I also thank Pengfei Lin for his diverse kinds of support and his superb skills in making professional and beautiful illustrations.

Next, I want to acknowledge all of my friends from all over the world, for their priceless inspiration and advice about my work. Especially, I must mention Xana Wu, a mathematician from the University of Queensland, for her ingenious construction and solid mathematical support to make my last work, zk-PoDI, possible.

Last but not least, I would like to convey my deepest love and gratitude to my parents and other family members. They have been my constant source of strength and support throughout my long and arduous Ph.D. journey. I dedicate this dissertation to them.

Dec 2023

Contents

Abstract **ii**

Acknowledgements **vi**

List of Figures **xiii**

List of Tables **xiv**

1 Introduction **1**
1.1 City-scale CAV behavior simulation 4
1.2 Proof of observation in cooperative perception 5
1.3 Preventing the Sybil attacks caused by simultaneous usage of pseudonyms 7
1.4 Contributions 8

2 Cooperative ITS and Its Security Concerns **11**
2.1 Recent Development and Deployment of Self-driving Vehicles . 11
2.2 Cooperative ITS and Vehicle to Everything 16
2.3 Security Requirements and Threats of C-ITS 18
2.4 Misbehavior Detection 19
2.5 Digital Signature and ECDSA 22
2.6 Zero-Knowledge Proofs 22
2.7 Pseudonyms in C-ITS 24
2.8 Sybil Attack and Its Countermeasures 25

3 City-scaled CAV Behavior Simulator **26**
3.1 Design 26
3.1.1 Scenario and Simulators 26
3.1.2 Sandbox and Different Types of Vehicles 28
3.1.3 Flow Inside the Vehicles 30
3.1.4 Metric Collection 31
3.2 Implementation Details 31

3.2.1 Programming Language and Its Implementation 31
3.2.2 PositionManagerV2 32
3.2.3 PerceptionSimulatorV3 33
3.3 Verification 37
3.4 Application 38
3.5 Summary 40

4 Zero-knowledge Proof of Traffic 42
4.1 Problem Statement, Assumptions and Approaches 42
4.1.1 Problem Statement 42
4.1.2 Assumptions 43
4.1.3 Approaches 44
4.2 Zero-Knowledge Proof of Traffic 44
4.2.1 Problem Conversion 45
4.2.2 Zero-Knowledge Proof of Shared Secret 45
4.2.3 Integrating zk-PoT into the ETSI CPS standards . . . 49
4.3 Evaluation 51
4.3.1 Parameter and Metric Settings 52
4.3.2 Manhattan Scenario: Bootstrap and Convergence . . . 53
4.3.3 LuST scenario: City-scaled Long-term Experiments . . 54
4.4 Threat Analysis 60
4.4.1 Brute-Force Attack and Dictionary Attack 60
4.4.2 Location Privacy Against Tracking 61
4.4.3 Spam Attack 61
4.4.4 Signal Jamming Attack 62
4.4.5 Collusion Attack and Sybil Attack 62
4.5 Summary 63

5 Zero-Knowledge Proof of Distinct Identity 64
5.1 Problem Statement, Assumptions and Requirements 64
5.1.1 Problem statement 65
5.1.2 Assumptions 66
5.1.3 Requirements 66
5.2 Mathematical Primitives 67
5.2.1 Diophantine Equation and its Integer Solution 67
5.2.2 zk-SNARKs 69
5.3 Proposed Method 71
5.4 Discussion 77
5.4.1 Performance Analysis 77
5.4.2 Threat Analysis 79
5.5 Summary 79

6 Conclusion **81**

Bibliography **84**

Publications **95**

List of Figures

1.1 General structure of CPS 6

2.1 QCraft Robobus 15
2.2 Meituan's driverless delivery vehicles 16
2.3 Taxonomy of MBD methods 20

3.1 Architecture of Flowsim 27
3.2 Example of defining vehicle types 28
3.3 Relevant points and corresponding angles in the camera coordinate. Green lines: projections of corners of vehicles; Red lines: projections of the rear numberplate; β: heading angle related to the camera position; δ_1, δ_2: argument angles of corners of vehicles; ρ_1, ρ_2: argument angles of the rear numberplate. 35
3.4 Example figure types generated from collected metrics 39
3.5 Number of vehicles and processing time breakdown 40

4.1 Procedure of a Zero-Knowledge Proof of Traffic 44
4.2 Overview of Zero-knowledge Proof System 46
4.3 Structure of the extended part of collective perception message (CPM) for zk-PoT 49
4.4 Block Diagram of Original v.s. Extended CPS 51
4.5 Manhattan scenario 54
4.6 Accumulated available perceived objects (N_a) in Manhattan scenario 55
4.7 LuST scenario with the position of the E44 and the roundabout 56
4.8 Number of active vehicles in the LuST scenario 57
4.9 Number of vehicles in Luxemburg city at different times 57
4.10 Verification ratio in Luxemburg city at different times 58
4.11 Vehicles and lane occupancy near the E44 roundabout at 08:42 59
4.12 Time-to-Verification (TTV) distribution 60
4.13 Bandwidth consumption per vehicle 61

5.1 Arithmetic circuit overview 73
5.2 Protocol overview 76
5.3 Proof and verification time of zk-PoDI 78

List of Tables

2.1 SAE Automation Levels 12

3.1 Implemented vehicle types in Flowsim 29

4.1 Common Parameter Settings 52

Chapter 1

Introduction

Road transportation has been one of the most essential services for human mobility since ancient times. The development of roads and vehicles has played a pivotal role in connecting people, facilitating trade, and enabling the exchange of ideas and cultures. Over the centuries, we have witnessed remarkable advancements in road transportation, from horse-drawn carriages and steam-powered locomotives to the modern automobiles we see today. However, the future of road transportation holds even more revolutionary prospects with the advent of autonomous driving technology.

Intelligent Transportation Systems (ITS) represent a paradigm shift in the domain of transportation, integrating advanced technologies to enhance the efficiency, safety, and sustainability of transportation networks. These systems leverage cutting-edge information and communication technologies to optimize the utilization of infrastructure, vehicles, and other resources within the transportation ecosystem. The overarching goal of ITS is to create a smart and interconnected transportation network capable of mitigating traffic congestion, reducing environmental impact, and enhancing overall mobility.

One significant component of ITS is Cooperative Intelligent Transportation Systems (C-ITS), a subset that emphasizes cooperative interactions among various entities within the transportation system. C-ITS leverages communication technologies to facilitate real-time information exchange between vehicles, infrastructure, and other relevant stakeholders. The integration of communication in C-ITS plays a pivotal role in enabling vehicles to communicate with each other and with the surrounding infrastructure, contributing to the creation of a safer and more efficient transportation environment.

The importance of communications in C-ITS cannot be overstated. Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communications enable

vehicles to share critical information, such as location, speed, and status, in real-time. This exchange of data empowers vehicles to make informed decisions, enhancing safety through collision avoidance and optimizing traffic flow. Moreover, C-ITS can provide drivers with timely and context-aware information, such as traffic conditions, road hazards, and optimal routes, leading to a more responsive and adaptive transportation system.

Standardization is a key facet of C-ITS implementation, ensuring interoperability and compatibility among diverse systems and devices. Standardized communication protocols and data formats are essential to facilitate seamless information exchange between different components of the transportation ecosystem. Organizations and standardization bodies, such as the European Telecommunications Standards Institute (ETSI) and the Institute of Electrical and Electronics Engineers (IEEE), play a crucial role in developing and promoting these standards to establish a unified framework for C-ITS deployment.

The deployment of C-ITS raises pertinent concerns regarding privacy and trust, as the extensive exchange of sensitive information in realtime introduces potential vulnerabilities that must be carefully addressed. One primary privacy concern revolves around the collection and dissemination of personal data through V2V and V2I communications. In C-ITS, vehicles share information such as location, speed, and trajectory, which, if mishandled, could compromise the privacy of individuals. Striking a balance between the need for information exchange and safeguarding individual privacy becomes paramount in the design and implementation of C-ITS.

On the other hand, trustworthiness is also a great concern of C-ITS, as trust issues may arise in cooperative perception, which relies on V2X communication to share local perception information among vehicles and infrastructure nodes. These issues include ensuring the security and privacy of the shared information from malicious attacks, evaluating the quality and credibility of the information and its sources, and balancing the trade-off between the benefits and costs of sharing information. Therefore, various mechanisms and techniques are needed to protect, assess, and motivate the C-ITS participants to cooperate in a trustworthy manner.

Among them, the credibility of the information shared in C-ITS is crucial for the safety and efficiency of road transport. However, it is challenging to verify the authenticity and accuracy of the information, as different sources may have different levels of trustworthiness, reliability, and consistency. Moreover, some sources may intentionally or maliciously provide false or misleading information, which could compromise the performance and security of the C-ITS, and this consequentially may lead to potential traffic jams and accidents.

The identification and mitigation of cyber-attacks within C-ITS constitute the primary objectives of misbehavior detection (MBD). This proactive undertaking encompasses two distinct categories: data-centric approaches and node-based approaches. Data-centric methodologies hinge upon plausibility and consistency measures to discern potential cyber threats. However, a noteworthy limitation of this approach lies in its capability to solely detect pre-defined misbehaviors, rendering it susceptible to both false positives and false negatives. Conversely, node-based approaches rely on the historical behaviors of a given vehicle or its reputation within the network of interconnected vehicles to assess the likelihood of dishonest conduct. Despite their potential, these approaches exhibit certain drawbacks. The dependency on past behavior introduces a notable latency in the establishment and regular updating of trust assessments. Consequently, this latency may impede the timely detection and response to emerging misbehaviors, thus compromising the overall efficacy of node-centric misbehavior detection strategies in the realm of C-ITS cybersecurity.

In recent years, as blockchain and cryptocurrencies have attracted more and more attention [Nak08], a completely different approach has also entered the vision of researchers in various fields. Technologies such as distributed ledger, smart contract, decentralized finance, web3, and zero-knowledge proofs are applied in many domains besides the original finance, including healthcare [AK23], education [LAB21], internet of things [PLH+21], logistics [RRST21], smart cities [MPA+22], government [VS22].

Unlike the traditional assumption that most people are honest, a.k.a. the honest majority assumption, these new methods share the common feature of assuming that everyone has dishonest motives, and instead use the game theory between different users to make dishonesty unprofitable. Specifically, in these techniques, the consensus among the parties is based on trusting only facts (and mathematical constructs based on those facts). Such a fact-based approach ensures the credibility of the data and the absence of disputes among the parties. This certainty is beneficial for cooperative ITS, because human lives are at stake, and many systems require strong consistency to ensure the reliability of judgments.

Especially zero-knowledge proofs, which are a cryptographic technique that allows one party to prove to another that a statement is true, without revealing any information beyond the validity of the statement, to protect the private data of the prover. Zero-knowledge proofs can be used to enhance the privacy and security of various applications, such as authentication, blockchain, and voting systems.

Following this fact-based paradigm, we leverage zero-knowledge proofs as a technique to enable vehicles to prove their identity and their perception of

other vehicles in a privacy-preserving manner. Furthermore, other vehicles can validate these proofs without depending on any other mechanisms and without having access to the ground truth.

1.1 City-scale CAV behavior simulation

With recent technology advancements, connected autonomous vehicles (CAVs) have emerged as a promising solution to enhance road safety, efficiency, and convenience. CAVs are vehicles equipped with advanced sensing, communication, and decision-making capabilities that enable them to operate autonomously and interact with their surroundings, including other vehicles, infrastructure, and pedestrians.

Comprehensive experiments and simulations are necessary to evaluate the performance and behavior of CAVs. However, existing simulators often lack the necessary features and flexibility to accurately represent the behavior of CAVs' on-board units (OBUs). Traffic simulators, such as SUMO [LBBW+18] focus on simulating traffic flow, whereas network simulators, such as NS-3 [RH10] and OMNET++ [Var10] primarily model wireless radio communication and packet delivery. VANET libraries and frameworks, such as artery [ROG19] and veins [SGD11], provide the implementation of complete protocol stacks that are useful to further investigate the protocols within the network simulators. Perception and localization simulators, such as Carla [DRC+17] and LGSVL [RST+20], provide realistic 3D environments, but they are not designed specifically for evaluating CAV behavior. OpenCDA [XGH+21] is a cooperative driving automation simulator enabling a full procedure of perception, localization, planning, and actuating. However, it only implements pre-defined cooperative driving models and is incapable of networking. Kinematics simulators, such as MATLAB and Simulink, can evaluate the consequences of planning decisions. However, they lack the capability to analyze complex CAV behaviors in a realistic context. Thus, a new simulator that focuses on the behavior and data flow of CAVs' OBUs is urgently required. Such a simulator should provide a city-scale experimental environment, offer high performance for large-scale scenarios, be extensible to accommodate various modules and functionalities, and provide flexibility for customizing and evaluating CAV behaviors. Additionally, reproducibility is another crucial requirement for ensuring that experiments can be repeated by other researchers to verify the results.

To close the gap among the simulators, we proposed Flowsim, a novel simulator designed specifically to meet the requirements for evaluating CAV behavior and data flow. Flowsim offers a modular and extensible architec-

ture that enables the analysis of CAV behaviors in large-scale scenarios. By providing a customizable simulation platform, Flowsim aims to accelerate the development and evaluation of cybersecurity measures and protocols for CAVs, thereby ensuring their safe and secure deployment. In addition to its primary purpose of evaluating CAV behavior, Flowsim has a wide range of potential use cases, such as protocol evaluation, cybersecurity assessment, traffic optimization, and policy development. The extensible and playable open environment of Flowsim fosters the exploration of novel algorithms and strategies for CAVs, thereby promoting innovation and collaboration within the research communities.

1.2 Proof of observation in cooperative perception

Cooperative perception is a groundbreaking technology to change the way of driving. It can expand our field of vision and reduce blind spots by leveraging the sensors of other vehicles. The standardization of cooperative perception in the ISO [Int20] and the ETSI [ETS23a] demonstrates its significance in the field. The Intelligent Transportation System (ITS) station architecture standardized in the ETSI designates the collective perception service (CPS) as a crucial safety-related application protocol, as depicted in Fig. 1.1. The protocol utilizes collective perception messages (CPMs) to share information regarding the observed vehicles, such as the position, velocity, and other important information with other vehicles. Although the CPS standard has been published recently, multiple open-source and proprietary CPM implementations have already been published [AJN+23], which greatly facilitates further research and development.

The CPMs received from other vehicles can influence the decisions made by the vehicles. Thus, ensuring the authenticity of these messages is crucial for road safety, as the CPMs are generated by vehicles rather than a centralized and trusted authority [HSBL17, GGS04]. Unlike a centralized authority, individual vehicles may be incentivized to deceive other vehicles to maximize their own profits [LSS+08]. As an example, a malicious vehicle could simulate congestion on the road ahead by sending synthesized CPMs, prompting other vehicles to take an alternative route, to clear the way for themselves.

Numerous solutions have been suggested to detect and eliminate such fraudulent activities. One such solution is the Public Key Infrastructure (PKI), which has been standardized in the VANET [ETS22]. This standard

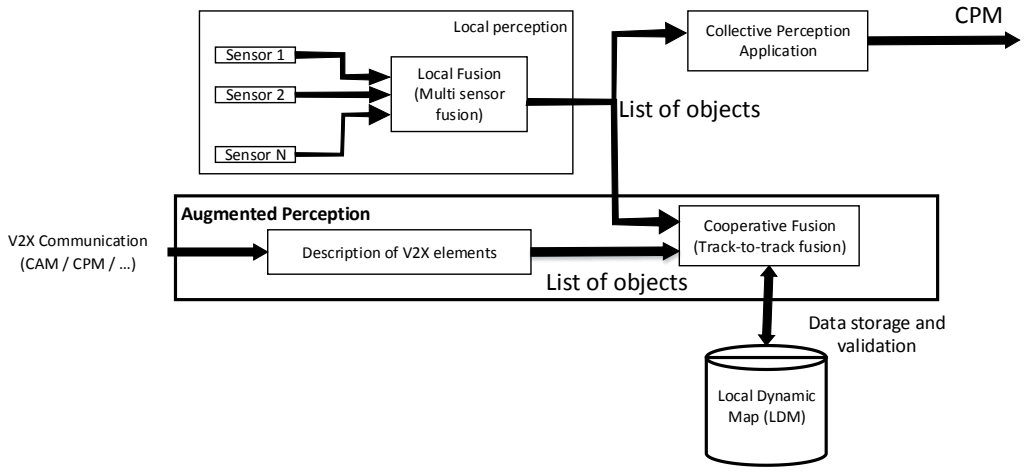


Figure 1.1: General structure of CPS

mandates that every message transmitted over the VANET must be signed using a secret key provided by law enforcement agencies (LEAs). Consequently, every node in the network is forced to obtain the keys from LEAs before sending any message.

Nevertheless, vehicles can still send fraudulent messages on purpose and sign them with their valid LEA-issued keys. Most existing security solutions for the Internet are impractical owing to VANET's highly decentralized and mobile nature. Various trust management methods have been proposed to overcome this problem [MZTC11, RPGH08, YCT+19, SS19].

To tackle the data authenticity issue in cooperative perception, we proposed a zero-knowledge proof (ZKP) based deterministic traffic cross-verification method called zero-knowledge Proof of Traffic (zk-PoT). Zk-PoT enables vehicles to prove the existence of vehicles they observed independently, then enables the remote parties to deterministically cross-verify the observations without any knowledge about the ground truth. Subject to the zero-knowledge property, the public cannot gain any extra information in the whole process, thus the privacy of the observed vehicles is preserved.

The proposed mechanism can enhance the security, efficiency, and verification latency while preserving the privacy of the target vehicles. Integrating the zk-PoT into existing cooperative perception standards such as ISO and ETSI requires minimal changes to the architecture, while still maintaining backward compatibility. Additionally, it can be used as a bootstrap method alongside existing trust management methods.

1.3 Preventing the Sybil attacks caused by simultaneous usage of pseudonyms

Pseudonyms, or temporary identifiers, are indispensable tools for safeguarding the privacy and security of communication in C-ITS. In C-ITS, pseudonyms are often presented as government-issued short-term digital certificates, which ensures the legality of the identities meanwhile protecting the privacy of the vehicles. ISO [Iso22] and ETSI [ETS22] standards require the pseudonym design to meet the unlinkability requirements [PH10], which essentially means the other party, except law enforcement agencies (LEAs) and the owner itself, should not be able to tell if two pseudonyms are owned by a same vehicle.

However, the legality and unlinkability of the pseudonym system make it naturally more vulnerable to Sybil attacks. The Sybil attack [Dou02] is a well-known security threat in distributed computer networks, particularly in the context of peer-to-peer and decentralized systems. A Sybil attack in C-ITS involves a malicious entity transmitting data using multiple pseudonyms simultaneously to pretend to be different vehicles. Such attacks can compromise the accuracy of critical information sharing, leading to misleading traffic updates, potentially causing accidents, or even exploiting vulnerabilities in traffic management protocols. The works that rely on voting mechanics or cross-verification are especially vulnerable to the Sybil attack, including our previous work zk-PoT [TJL+23], a zero-knowledge cross-verification mechanism for the existence of an observed vehicle. Multiple sybil-resist pseudonyms are proposed [MMZo21, FFB16, FWB15, LSL+21, AMH23] to solve this problem, but they either require a global database including a blockchain, or they require some prior knowledge such as the social graph, which are both considered unpractical in C-ITS because its locality and intermittent connectivity nature.

For vehicle networks, especially vehicular ad-hoc networks (VANETs), it is infeasible to assume the vehicles always have a global connection or a connection to the infrastructure (LEA or RSUs). Therefore, issuing the pseudonyms in advance and letting the vehicle preloading them becomes a more flexible and viable approach. Nevertheless, such a preloading mechanism will give extra difficulties for the LEA to handle the lifecycle of the pseudonyms. Specifically, the mechanism must result in the situation that multiple pseudonyms are valid simultaneously.

In C-ITS, every pseudonym is a valid ID, a vehicle can have multiple pseudonyms at a time, and the owner of a pseudonym is not distinguishable by the public. These three features facilitate the simultaneous usage of pseudonyms, i.e. conducting a Sybil attack. On the other hand, if a

pseudonym system enables the public to determine if the pseudonyms are simultaneously used, it essentially breaks the unlinkability property, which is not a viable approach either.

This leaves us with a paradox: how can the vehicle give out just the right amount of knowledge, to convince the others, but not compromise the unlinkability?

We observed that, in C-ITS, the vehicles do not need information that is too old or too distant. Consequently, they do not care if a nearby vehicle also appears in, say, 1000 km away. In other words, among various conflicting demands in C-ITS, compromising the global Sybil-resistant has the least impact on the entire system. Therefore, in this work, we gave up the pursuit of global Sybil-resistant and instead attempted to address the issue of local Sybil-resistant. In other words, a vehicle no longer needs to prove that it is different from every vehicle, but only differs from the given surrounding vehicles. This is called "**the locality assumption**."

Based on the locality assumption, we proposed a novel method called zero-knowledge Proof of Distinct Identity (zk-PoDI), an extension to the pseudonym systems. zk-PoDI enables a vehicle to prove that it is *not* the owner of a given pseudonym, while not breaking the unlinkability property of the pseudonyms. Utilizing the zk-PoDI, a vehicle can prove it is not another vehicle whenever needed, i.e. it is not potentially conducting a Sybil attack, thus its identity is legit and its shared information is trustworthy. Zk-PoDI has very few assumptions underlying pseudonyms and thus can be applied to a vast amount of pseudonym systems as an extension.

1.4 Contributions

As we have previously discussed, trust-based approaches are gaining momentum in both theoretical and practical aspects. This gives the C-ITS academia a new perspective and an indispensable technology, zero-knowledge proof. With this tool, we have made significant contributions to improving C-ITS in various aspects. We have achieved this by converting real-world problems into mathematical problems, finding the appropriate mathematical constructions, i.e., the zero-knowledge proof protocols, to solve the mathematical problem, proposing concrete systems and protocols to solve real-world problems based on the mathematical solutions, and finally, evaluating them with proof-of-concepts (PoCs) and comprehensive experiments.

The contributions in this dissertation are as follows:

1. **Building a city-scaled lightweight vehicle behavior simulator.** To achieve the precise evaluation of CAV behavior in large-scale en-

vironments, we proposed Flowsim, a lightweight simulator written in Python. With a modular and extensible architecture, Flowsim allows researchers to analyze CAV behaviors, communication protocols, cybersecurity vulnerabilities, traffic management strategies, and policies in customizable scenarios. Through experiments based on realistic traffic scenarios, Flowsim demonstrates its effectiveness, offering a valuable tool for CAV development and deployment. Notably, the simulator's 1,500 lines of code in pure Python emphasize readability and modifiability, and its open-source availability on GitHub under the MIT license encourages collaboration and contributions from the research community.

2. **Introducing a privacy-preserving mechanism for vehicles to prove and cross-verify their observations.**
   We introduced the zero-knowledge Proof of Traffic (zk-PoT) as a novel approach to address authenticity challenges in cooperative perception for connected automated vehicles in intelligent transportation systems. zk-PoT utilizes cryptographic proofs for traffic observations, allowing deterministically cross-verified proofs for the same vehicle without relying on ground truth or predefined rules. The method proves resilient against various attacks and can be seamlessly integrated into existing standards like ISO and ETSI, showcasing its potential for enhancing security and privacy while maintaining compatibility with established frameworks.

3. **Proposing a zero-knowledge proof to block Sybil attacks by preventing simultaneous use of pseudonyms.**
   We introduced the zero-knowledge Proof of Distinct Identity (zk-PoDI) to address a type of Sybil attack by simultaneously using multiple pseudonyms in C-ITS. The zk-PoDI acts as an extension of most existing pseudonyms, enabling the vehicles to prove that it is not the owner of other pseudonyms that appeared in the surrounding area. This method can convince the other vehicles of such a statement without compromising the unlinkability of the pseudonym, maintaining the location privacy of the vehicle. The method features low computational cost and negligible communication cost. It shows strength against various attacks.

The remainder of this dissertation is organized as follows.

Chapter 2 raised several concerns about the security issues in ITS, introduced various works and techniques related to our work, and discussed their limitations.

Chapter 3 proposed Flowsim, a city-scale extensible lightweight vehicle behavior simulator. Then, based on Flowsim, several trust and data authenticity issues are explored and solutions to address each issue are proposed.

Chapter 4 proposed the zk-PoT, a privacy-preserving protocol for vehicles to prove and cross-verify their observation to the traffic.

Chapter 5 proposed the zk-PoDI, an extension of pseudonym systems for vehicles to prove their distinctiveness, to address the Sybil attacks.

Finally, chapter 6 wrapped up the dissertation by summarizing the contributions and pointing out some future directions.

Chapter 2

Cooperative ITS and Its Security Concerns

In this chapter, we will explore the market status and technological development of self-driving vehicles from the perspectives of three leading countries. Then, we will shift our perspective from industry to academia, and have a systematic introduction of C-ITS, V2X, and its security concerns. Finally, we will introduce the related work addressing the cybersecurity threats of C-ITS.

2.1 Recent Development and Deployment of Self-driving Vehicles

Society of Automotive Engineers (SAE) proposed a standard taxonomy and framework for describing the degree of driving automation in on-road motor vehicles [Soc21]. The SAE Automation Levels are widely adopted by the industry, regulators, and consumers to provide clear and consistent terminology and definitions for different levels of automation. The SAE Automation Levels range from Level 0 (no automation) to Level 5 (full automation), based on the extent to which the driving task is performed by the human driver or the automated system, as shown in Table 2.1. The SAE Automation Levels also specify the conditions and requirements for each level of automation, such as the operational design domain, the object and event detection and response, and the fallback performance. The SAE Automation Levels are regularly updated to reflect the latest developments and challenges in the field of automated driving.

In a significant advancement for autonomous vehicle technology, Japan has achieved a groundbreaking milestone with the approval of an Automated

Table 2.1: SAE Automation Levels

Lv.	**Name**	**Description**	**Driving**	**Monitoring**	**Fallback**
0	No Automation	Full-time performance by the driver of all aspects of the dynamic driving task, even when enhanced by warning or intervention systems	Driver	Driver	n/a
1	Driver Assistance	An ADAS of either steering or braking/acceleration subsystems that are not designed to coordinate with each other	Driver	Driver & System	Driver
2	Partial Automation	An ADAS that has the capability to control both steering and braking/acceleration subsystems and is designed to coordinate with each other	Driver	Driver & System	Driver
3	Conditional Automation	An ADS that performs all aspects of the dynamic driving task with the expectation that the driver will respond appropriately to a request to intervene	Some Driving Modes	System	Driver
4	High Automation	An ADS that performs all aspects of the dynamic driving task, even if a human driver does not respond appropriately to a request to intervene	Some Driving Modes	System	System
5	Full Automation	An ADS that performs all aspects of the dynamic driving task under all roadway and environmental conditions that can be managed by a human driver	All Driving Modes	System	System

Driving Vehicle as an Automated Operation Level 4 System for the first time in the country. This approval marks a pivotal moment in the evolution of self-driving vehicles, setting a precedent for future developments in the field.

The Ministry of Economy, Trade and Industry (METI) and the Ministry of Land, Infrastructure, Transport and Tourism (MLIT) have been collaborating on the "Project on Research, Development, Demonstration and Deployment (RDD&D) of Automated Driving toward Level 4 and its Enhanced Mobility Services," also known as the "RoAD to L4" project. This initiative aims to facilitate the realization of automated driving mobility services, and it has been underway since FY2021.

On March 30, 2023, the automated driving vehicles designated for use in Eiheiji Town, Fukui Prefecture, received approval under the Road Transport Vehicle Act as a Level 4 automated operation system. This classification signifies that the system can perform all driving operations independently within a limited area and under specific conditions, without the need for a driver on board.

The approval process involved a series of validation tests conducted as part of the RoAD to L4 project. These tests were designed to assess the technologies and services related to automated driving vehicles and remote monitoring systems. The National Institute of Advanced Industrial Science and Technology (AIST), along with a consortium of private entities, played a crucial role in submitting the request for approval to the Chubu District Transport Bureau, MLIT.

This achievement aligns with Japan's "Grand Design and Action Plan for a New Form of Capitalism and Follow-ups," which set forth the objective of initiating automated driving mobility services operated solely by remote monitoring around FY2022. The successful approval of the Level 4 system is a testament to the country's commitment to advancing its automotive industry and embracing the potential of autonomous technology.

The implications of this development are far-reaching, as it not only enhances the prospects for automated driving in Japan but also contributes to the global discourse on the integration of autonomous vehicles into society. It represents a step towards a future where transportation is safer, more efficient, and more accessible, thanks to the innovative strides made in automated vehicle technology.

In recent years, China has been a pioneer in the promotion of autonomous driving technologies, implementing a series of supportive policies. The country is targeting breakthroughs in key technologies such as smart road infrastructure and cooperative systems between vehicles and infrastructure by 2025 [Sta21]. The government is integrating various innovative resources to foster advancements in autonomous driving technologies, including inte-

grated sensing systems and smart infrastructure. It also supports the application of autonomous technologies in cargo and passenger transportation services and encourages the piloting of new business models like car-sharing and smart parking services. To create a conducive policy environment, the government is refining regulations on road tests for autonomous vehicles. These measures have been welcomed by industry insiders, instilling confidence in the sector's growth.

China is advancing rapidly in the development of self-driving vehicles, thanks to policy support. As of May 2023, over 15000 kilometers of roads have been opened for testing, with a cumulative test mileage exceeding 60 million kilometers and over 2500 test licenses issued. There are currently 696 companies in China providing services and products related to the autonomous driving market.

Apollo is a leading autonomous driving technology platform, was launched by Baidu in April 2017. The platform provides an open, secure, and reliable solution that supports all major features and functions of an autonomous vehicle. One of the key services offered by Baidu Apollo is the Apollo Go Robotaxi service. This service allows passengers to hail self-driving cars via a mobile app. As of the end of Q2 2021, the Baidu Apollo autonomous driving service has provided more than 400,000 rides and driven more than 8.7 million miles. In January 2023, Apollo Go reached a major milestone by providing 1 million EV autonomous rides. Furthermore, Baidu Apollo expanded its fully driverless ride-hailing service to Shenzhen after Beijing and Wuhan. This expansion demonstrates Baidu Apollo's commitment to advancing autonomous driving technology and promoting its widespread adoption.

QCraft is founded in 2019, is a world-leading self-driving technology company. The company focuses on the research and development of autonomous driving vehicles such as buses and ride-hailing cars. Their mission is to bring autonomous driving into real life. In March 2021, QCraft launched its third fleet of robobuses in Wuhan, China, as shown in Figure 2.1. The robobuses are equipped with Ouster lidar sensors, which help the bus eliminate blind spots with their wide field of view and high resolution. QCraft plans to have at least 100 autonomous buses on open roads in China by the end of this year.

Meituan is a Chinese online platform that offers various services, such as food delivery, travel booking, and group buying. One of its services is Meituan Autonomous Delivery (MAD), which uses self-driving vehicles to deliver goods to customers. Meituan has been developing and testing



Figure 2.1: QCraft Robobus

its driverless delivery vehicles since 2018. It has also obtained licenses to operate its vehicles on public roads in Beijing, Shanghai, and Shenzhen. Meituan's driverless delivery vehicles are small, electric-powered, and can carry up to 100 kilograms of goods as shown in Figure 2.2. They can navigate the roads, avoid obstacles, and communicate with customers. Customers can order goods on Meituan's app and choose the option of autonomous delivery. They can track their orders and open the vehicle's compartment with a QR code or a password. Meituan claims that its driverless delivery vehicles can lower the cost, time, and carbon emissions of delivery, and improve the efficiency, safety, and customer satisfaction of the service. Meituan also plans to expand MAD to more cities and scenarios in the future, such as grocery shopping, medical supplies, and campus delivery.

Pony.ai is a leading global autonomous driving technology company that develops full-stack solutions for robotaxi, robotruck, and personally owned vehicles. Pony.ai has established a strategic joint venture with Sany Heavy Truck, a subsidiary of Sany Heavy Industry, China's number one heavy equipment manufacturer and the third largest heavy equipment manufacturer globally, to develop and mass produce Level



Figure 2.2: Meituan's driverless delivery vehicles

4 (L4) robotrucks for the Chinese market. L4 robotrucks are capable of driving themselves on highways and urban roads without human supervision, under specific conditions. The joint venture plans to start small-scale deliveries of the robotrucks in 2022 and 2023, and mass production of the L4 truck will commence in 2024, with an annual production of around 10,000 trucks within a few years. The robotrucks will be built on Sany's new electric vehicle (EV) truck platform, and will be powered by Pony.ai's Autonomous Driving Controller (ADC), which is based on the NVIDIA DRIVE Orin chip. The robotrucks will also include a mix of new energy vehicles (NEVs) and fuel-based vehicles, with the aim to increase the proportion of NEV trucks in the future.

2.2 Cooperative ITS and Vehicle to Everything

Cooperative ITS, or C-ITS, is an extension of self-driving cars, which are equipped with sensors and software that enable them to navigate roads with-

out human intervention. C-ITS takes this a step further by allowing vehicles to communicate with each other and with the infrastructure around them, such as traffic lights and road signs. This communication enables real-time data exchange, which can help avoid accidents, reduce congestion, and improve overall traffic flow.

Vehicle-to-Everything (V2X) communication is a crucial component of C-ITS, facilitating real-time data exchange between vehicles and other entities in the transportation ecosystem. This bidirectional communication allows for the sharing of information related to traffic conditions, road hazards, and other critical data, contributing to enhanced safety, traffic efficiency, and overall mobility in smart cities. Common communication types of V2X are listed below:

V2V (Vehicle-to-Vehicle): Direct communication between vehicles enables the exchange of information about speed, position, and status, enhancing situational awareness and enabling cooperative maneuvers.

V2I (Vehicle-to-Infrastructure): This communication type involves vehicles interacting with roadside infrastructure, such as traffic lights and road signs, providing valuable information to optimize traffic flow and enhance safety.

V2P (Vehicle-to-Pedestrian): V2P communication allows vehicles to exchange data with pedestrians' devices, such as smartphones, to improve safety by notifying drivers of pedestrians in their vicinity and vice versa.

V2N (Vehicle-to-Network): Vehicles communicate with the transportation network, allowing them to access real-time traffic information, updates, and services, enhancing overall traffic management and efficiency.

V2X communication relies on standardized messages that convey crucial information among vehicles and infrastructure in C-ITS. These messages play a pivotal role in enabling seamless communication and cooperation, fostering safer and more efficient traffic environments. Common message types in V2X are listed below:

CAM (Cooperative Awareness Message): a fundamental message in V2X communication, providing essential vehicle information such as position, speed, heading, and vehicle type. It contributes to situational awareness and supports cooperative maneuvers between vehicles.

DENM (Decentralized Environmental Notification Message): enables the exchange of information about environmental conditions, road hazards, and events. It supports real-time notifications to other vehicles and infrastructure, enhancing overall safety and traffic management.

CPM (Cooperative Perception Message): involves the sharing of perception data among vehicles, allowing them to collaboratively build a more comprehensive understanding of their surroundings. This includes information about objects, obstacles, and environmental conditions, contributing to enhanced safety and autonomous driving capabilities.

2.3 Security Requirements and Threats of C-ITS

The distributed and ad-hoc nature of C-ITS exposes it to a multitude of cybersecurity threats. These threats can compromise the safety and reliability of communication and data exchange, leading to serious consequences ranging from minor disruptions in traffic flow to major accidents. By understanding and mitigating the potential threats, we can ensure the resilience and reliability of our transportation systems, safeguarding the benefits that C-ITS can bring to our society. Common possible attacks are listed below:

Denial of Service (DoS): Attackers attempt to disrupt the availability of C-ITS services by overwhelming communication channels or system resources, rendering them inaccessible to legitimate users.

Jamming: Jamming attacks involve intentional interference with wireless communication, disrupting the exchange of messages between vehicles or between vehicles and infrastructure.

Flooding: Flooding attacks aim to overwhelm the communication network with a high volume of illegitimate messages, causing congestion and potentially disrupting normal operations.

Sybil Attacks: In Sybil attacks, a malicious entity assumes multiple false identities, deceiving the system into accepting illegitimate messages or creating confusion about the actual number of participating entities.

False Data Injection: Adversaries may inject false or misleading information into the C-ITS network, leading to incorrect decisions by vehicles or infrastructure components. This can compromise safety and traffic management.

Replay Attacks: In replay attacks, previously recorded messages are maliciously retransmitted to the system. This can lead to unauthorized access, manipulation of data, or disruption of normal system operation.

Man-in-the-Middle (MitM): In MitM attacks, an attacker intercepts and potentially alters the communication between two parties, gaining unauthorized access to sensitive information or injecting malicious content.

Eavesdropping: Unauthorized interception of communication to gather sensitive information. This can compromise user privacy and expose critical data to malicious actors.

2.4 Misbehavior Detection

Misbehavior detection (MBD) is a critical aspect of securing C-ITS against malicious activities and ensuring the reliability of communication and data exchange. In C-ITS environments, where vehicles and infrastructure interact to improve safety and traffic efficiency, the potential for misbehavior, intentional or unintentional, poses significant challenges.

MBD systems are designed to identify and mitigate various types of malicious actions, such as message manipulation, denial of service attacks, and deceptive behaviors that could compromise the effectiveness and safety of the entire system. These detection mechanisms play a vital role in maintaining the integrity and trustworthiness of C-ITS.

In the realm of C-ITS, MBD strategies can be broadly categorized into node-centric and data-centric approaches. Node-centric approaches focus on evaluating the trustworthiness of individual nodes within the C-ITS network. This involves assessing the historical behavior of entities, establishing trust levels, and identifying anomalies in their actions. Node-centric approaches can be sub-categorized into three types:

Trust-based MBD relies on establishing and evaluating the trustworthiness of individual nodes within the C-ITS network. Entities with a history of reliable behavior are trusted, while those with suspicious activities may be flagged.

Voting-based MBD involves multiple nodes collectively assessing the behavior of a particular entity. Voting mechanisms involve nodes expressing their opinions on the legitimacy of actions, and a consensus is reached to identify potential misbehavior.

Behavior-based MBD focuses on analyzing the behavior patterns of nodes within the C-ITS ecosystem. Deviations from expected or normative behavior are flagged as potential misbehavior, allowing for proactive identification.

On the other hand, data-centric approaches concentrate on the content and consistency of the information exchanged among nodes. Consistency-based methods evaluate the coherence of data, identifying inconsistencies that may signal misbehavior. Node-centric approaches can also be sub-categorized into three types:

Consistency-based MBD assesses the coherence of data and messages exchanged among nodes. Inconsistencies, such as conflicting information from a node, may indicate misbehavior.

Plausibility-based MBD checks if a message sent is considered possible. It checks for a valid range of positions, speed, etc. with a set of predefined criteria or a machine learning model.

Hybrid MBD combines elements of both node-centric and data-centric approaches. It leverages a comprehensive assessment that considers both the behavior of nodes and the plausibility and consistency of exchanged data.

The choice between node-centric and data-centric approaches often depends on the specific characteristics of the C-ITS environment and the desired balance between assessing the behavior of entities and scrutinizing the integrity of the exchanged information. Figure 2.3 shows the taxonomy of MBD methods.

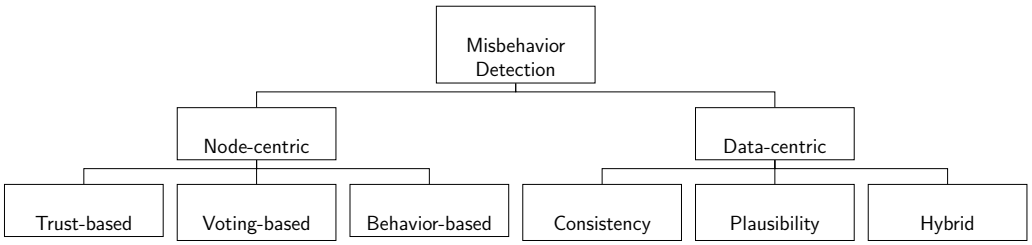


Figure 2.3: Taxonomy of MBD methods

Zacharias et al. [ZF18] proposed an autonomous node-based MBD system based on the local traffic density. This approach utilized multiple independent sensors to measure the traffic density and combined the evidence using

the Dempster rule of combination to detect misbehavior, particularly focusing on illusion attacks.

Al-Ali et al. [AAAMA20] proposed a blockchain-based collaborative approach to validate traffic events and authenticate vehicles in VANET. It utilizes the reputation scores, proof of authority (PoA) and proof of event (PoE) consensus algorithms, as well as mutual authentication between vehicles and roadside units (RSUs) to improve the event validation accuracy and detect internal attackers.

Numerous data-centric methods have also been proposed over the years. Ghosh et al. [GVG+10] proposed an autonomous MBD scheme for the post crash notification (PCN) application. It involves identifying the root cause of misbehaviors by constructing a cause tree and using logical reduction. This scheme achieves adequate detection rates and exhibits robustness to small errors in the parameter estimation.

Lo et al. [LT07] introduced a security threat in VANETs called the illusion attack, where an adversary broadcasts false traffic warning messages based on the current road conditions, thereby creating an illusion for nearby vehicles. This illusion can manipulate the drivers' behaviors, leading to car accidents, traffic jams, and decreased VANET performance. The authors proposed an autonomous data-centric security model called the plausibility validation network (PVN) to address this issue by cross-verifying the plausibility of incoming message fields. The problem they were aiming to solve is very similar to our problem, as we also utilize cross-verification, due to which the requirements are very similar.

Ercan et al. [EAM22] proposed a machine learning-based Intrusion Detection System (IDS) to detect position falsification attacks in VANETs. The IDS utilizes three new features corresponding to the sender's position, along with the k-nearest neighbor (kNN) and random forest (RF) classification algorithms. The results demonstrate that the proposed mechanism outperforms the existing approaches in terms of classification performance and computation time.

Kristianto et al.[KLH23] proposed a semi-supervised federated learning MBD system for V2X communications. This model addresses the challenges of limited labeled data and bandwidth consumption by leveraging semi-supervised learning and federated learning approaches. The experimental results show that the model achieves high performance, outperforming centralized supervised learning methods regarding the F1-score, recall, and reduced bandwidth utilization.

Although all the above works are proposed, the MBD problem still cannot be solved perfectly in all situations. Data-centric methods require a significant amount of data, presenting higher channel occupation, packet loss, and

increased latency. Furthermore, feasibility evaluations are the only options available in cases where the ground truth data are missing or difficult to obtain. In entity-centric methods, data correctness is still a major problem owing to the presence of attackers and sensor limitations.

2.5 Digital Signature and ECDSA

A digital signature is a mathematical scheme that uses asymmetric cryptography to verify the authenticity of digital messages or documents. The Digital Signature Algorithm (DSA) is a standard of digital signature based on the discrete logarithm problem. ECDSA [JMV01] is a variant of the DSA that employs elliptic curve cryptography, which is widely used in VANET and V2X applications, providing authentication, integrity protection, and privacy enhancements [HMP22, SKVG+23, KKSS19]. ECDSA also serves essential roles in the existing ETSI ITS standards [ETS22, ETS15]. Many experiments and analyses regarding ECDSA's performance and impact on V2X communication are carried out [CZSL19, HEGK20, PSFM20, Pet09, RSF+19], providing a comprehensive understanding about ECDSA's performance in the context of ITS and VANETs.

2.6 Zero-Knowledge Proofs

Zero-knowledge proofs were introduced by S. Goldwasser, S. Micali, and C. Rackoff in 1989 [GMR89]. They also introduced interactive proof systems in the same article. Such systems allows one party, called the prover, to convince another party, called the verifier, that a specified statement is true, without disclosing any information beyond the statement's validity.

Zero-knowledge proofs must satisfy the following three conditions:

Completeness: If the statement is true, an honest verifier should be convinced of the prover's claim.

Soundness: If the statement is false, there should be only a small probability that the verifier erroneously accepts the dishonest prover's claim.

Zero Knowledge: Throughout the process, the verifier should learn nothing except the truth of the statement.

While ZKP systems are widely used in cryptocurrency protocols such as Zerocoin [MGGR13] and Zerocash [BSCG+14], they are still relatively new in other fields of study. Recently, there has been a growing focus on utilizing

zero-knowledge proofs to enhance the security and anonymity of C-ITS.

McEntyre et al. [MK22] present a privacy-preserving electronic toll collection (ETC) protocol for V2X communications, utilizing a ZKP challenge set to ensure security while addressing the limitations of embedded technology. The protocol achieves toll verification without disclosing sensitive subscriber information, such as GPS location, by generating unique randomized challenges localized to toll areas, thereby reducing the likelihood of false tolling.

Chaudhry et al. [CSA+21] introduce a secure communication framework for vehicle-to-healthcare everything (V2HX) communications in fog computing environments, utilizing a combination of ZKP and statistical fingerprinting (SF) protocols. The proposed framework enables vehicle authentication through ZKP and ensures secure communication between VANETs and healthcare enterprises through SF.

Rasheed et al. [RMHL20] present an Adaptive Group-based Zero Knowledge Proof Authentication Protocol (AGZKP-AP) for VANETs, addressing privacy concerns in authentication. The protocol enables anonymous authentication, distributed privilege control, and customizable privacy settings for users while minimizing the disclosure of authentication parameters, enhancing user privacy in VANETs.

Li et al. [LMGN23] introduce an aggregated zero-knowledge proof and blockchain-based authentication system for privacy-preserving identity verification in autonomous truck platooning, addressing security and privacy concerns. The proposed approach enhances security, provides fast performance, and ensures data integrity while allowing truck companies to define access control policies. Experimental results on the Hyperledger platform demonstrate the system's feasibility for real-world truck platooning applications.

In [XLJM22], the authors introduced a novel and efficient anonymous authentication approach for the Internet of Vehicles (IoV). This method employs zero-knowledge proofs and elliptic curve cryptography to improve user privacy and service efficiency, achieving robust anonymity, authenticity, and traceability while surpassing traditional protocols in the IoV.

In the realm of wearable-based Collaborative Indoor Positioning Systems, privacy concerns prompted the proposal of a decentralized Attribute-based Authentication (ABA) protocol in [CMTSHG23]. This protocol addresses the limitations of existing protocols, offering heightened security and privacy for Collaborative Indoor Positioning Systems (CIPSs) within the expanding Internet of Things (IoT) applications landscape.

Additionally, in [WHM+23], the authors presented a lightweight zero-knowledge authentication scheme for IoT-embedded devices (LZIA) to overcome challenges related to computation, communication, and storage costs.

Experimental results demonstrate that LZIA outperforms existing schemes in terms of both security and performance by reducing computation, communication, and storage costs.

2.7 Pseudonyms in C-ITS

Most V2X applications rely on continuous collection and sharing of sensitive information such as detailed location information, which raises privacy concerns. To address this, a common approach is to use multiple temporary identifiers, called pseudonyms, to conceal the long-term identity of C-ITS vehicles. In C-ITS, pseudonyms of a vehicle are usually issued by the government as valid lawful identities.

In the literature, many works are proposing a new type of pseudonym to better adapt to the application scenarios of C-ITS. They also keep security in mind and show strength against various attacks.

Mundhe et al. [MYS+20] proposes an efficient Lightweight Ring Signature-Based Conditional Privacy-Preserving Authentication (RCPPA) scheme for VANETs. It employs pseudonym-based authentication and lattice-based ring signatures to ensure message authentication without revealing the real identity of vehicles. The scheme provides conditional privacy, allowing the disclosure of a vehicle's real identity by a trusted authority only in case of malicious actions, offering non-repudiation and security against unauthorized intruders. It shows unforgeability under an adaptive chosen-message attack.

Ali and Li [AL20] introduce an efficient Identity-based Conditional Privacy-Preserving Authentication (ID-CPPA) signature scheme for V2I communication in VANETs. Addressing the inefficiency in current schemes, the proposed ID-CPPA scheme utilizes bilinear maps and general one-way hash functions, supporting batch signature verification at RSUs. The scheme ensures security against adaptive chosen-message and adaptive chosen-identity attacks under the Inverse Computational Diffie-Hellman problem in the Random Oracle Model.

From the above examples, we can see that the current proposed works about pseudonyms do not directly have a built-in mechanism against the Sybil attacks. It is thus crucial to propose some mechanism for Sybil resistance or Sybil detection to extend those works.

2.8 Sybil Attack and Its Countermeasures

Sybil attacks are a type of malicious behavior that threatens the security and privacy of C-ITS. In a Sybil attack, an attacker creates multiple fake identities and uses them to broadcast false or misleading information to other vehicles or infrastructures. For example, an attacker can send fake traffic alerts to cause congestion or accidents or claim multiple services or resources that are intended for legitimate users. Sybil attacks are hard to detect and prevent, as the attacker can simultaneously use multiple valid pseudonyms to disguise as different vehicles. Therefore, effective Sybil detection or prevention mechanisms are needed to ensure the reliability and trustworthiness of C-ITS.

C-ITS is especially vulnerable to Sybil attack because of its pseudonym design. Many works, either Sybil-resist pseudonyms or the mechanisms helping other pseudonyms block Sybil attacks, were proposed.

Baza et al. [BNM+22] proposes a novel Sybil-attack detection scheme for C-ITS. The proposed scheme employs proofs of work and location, where each Road Side Unit (RSU) issues a time-stamped tag, and the vehicle creates a trajectory by combining consecutive tags, which serves as the vehicle's anonymous identity. A threshold signature mechanism prevents a single compromised RSU from generating fake trajectories. Additionally, vehicles solve computational puzzles using proof of work (PoW) algorithms, reducing the The paper includes extensive experiments and simulations, demonstrating the scheme's high detection rate against Sybil attacks with low false negatives and acceptable communication and computation overhead.

Trauernicht et al. [TB] introduces a concept for the long-term exclusion of Sybil attackers in C-ITS while preserving the privacy of benign stations. The proposed Sybil-attack Alternation Check (SAC) mechanism prevents the reuse of recently used Authorization Tickets, allowing the recording and reporting of attacks for certificate invalidation. The process ensures no false exclusions of benign stations and maintains privacy during misbehavior evaluation. Aligned with ETSI standards, the concept adds minimal complexity to the existing public key infrastructure. It is designed for practical industry implementation, with low overhead and minimal additional software features required.

Chapter 3

City-scaled CAV Behavior Simulator

To do a good job, an artisan needs the best tools. In this chapter, we will introduce our first work, Flowsim, a city-scaled extensible vehicle behavior simulator, to help inspect the behavior of C-ITS components inside vehicles, and evaluate the impact of our other works in the scale of a whole city. [1]

3.1 Design

This section provides an overview of the structure and the design of Flowsim, which is a multi-agent simulator with a modular and extensible architecture. Figure 3.1 illustrates the high-level structure of Flowsim, which can be divided into three main parts: simulators, sandbox, and vehicles within the sandbox.

3.1.1 Scenario and Simulators

As a simulation platform for evaluating the behavior of CAVs, Flowsim must provide agents with different types of information. Therefore, Flowsim integrates various simulation mechanics, including positioning, perception, and networking.

`PositionManager` establish a co-simulation with SUMO and update the positions of all agents from SUMO. Then, it provides the information to the

[1] Contents of this chapter have been published as the following article: **Ye Tao**, Ehsan Javanmardi, Jin Nakazato, Manabu Tsukada, Hiroshi Esaki. "Flowsim: A Modular Simulation Platform for Microscopic Behavior Analysis of City-Scale Connected Autonomous Vehicles." The 26th Edition of the IEEE International Conference on Intelligent Transportation Systems (ITSC 2023).

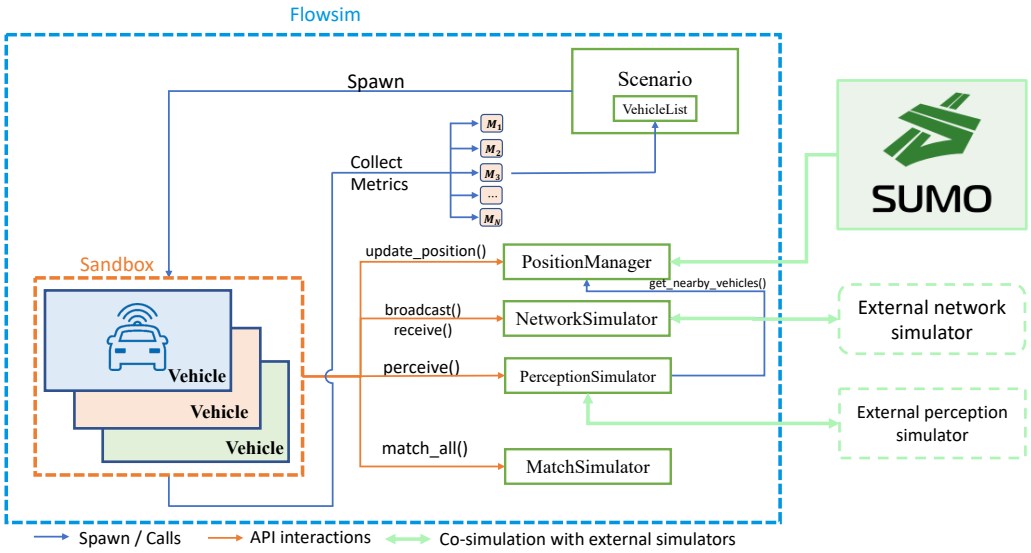


Figure 3.1: Architecture of Flowsim

relevant modules, such as the simulators and the vehicles. Moreover, the position manager records the ground truths for external queries.

`NetworkSimulator` simulates networking-related functions. Currently, it supports single-hop broadcast (SHB) in the ETSI GeoNetworking Standard [ETS20] as the only implemented networking feature.

`PerceptionSimulator` is a computer vision simulator. Instead of rendering realistic scenarios and vehicles and performing object detection algorithms on video feeds, it extracts positional information from objects. It simulates the projection from the ego vehicle's front camera, including FOV, occlusion, and filtering. Finally, it outputs fully visible and recognizable numberplates based on predefined criteria.

Meanwhile, `MatchSimulator` is a specially designed "non-standard" module in Flowsim. Generally, CAVs have two types of identity: numberplates like every vehicle, and an additional identity used for vehicle-to-everything (V2X) communication. While matching these two identities is an open research topic in the literature [MMT+23], Flowsim simplifies the simulation setup by internally maintaining a match table using ground truth data from other simulators and the `VehicleList`.

The main purpose of Flowsim is to analyze and simulate the behavior and data flow within and among vehicles. Currently, for simplicity and performance reasons, `NetworkSimulator`, `PerceptionSimulator`, and `MatchSimulator` are implemented from scratch. However, if a more precise evaluation is required, such as simulating channel collisions in high-density

traffic, the corresponding parts of Flowsim can be connected to external simulators.

Finally, the `Scenario` module handles the top-level information and controls the entire lifecycle of a simulation. It spawns vehicles and simulator modules, manages ground truth data, and collects metrics from all simulators and agents. By adopting a modular and extensible design, Flowsim provides a flexible and powerful simulation platform for analyzing CAV behavior and data flow.

3.1.2 Sandbox and Different Types of Vehicles

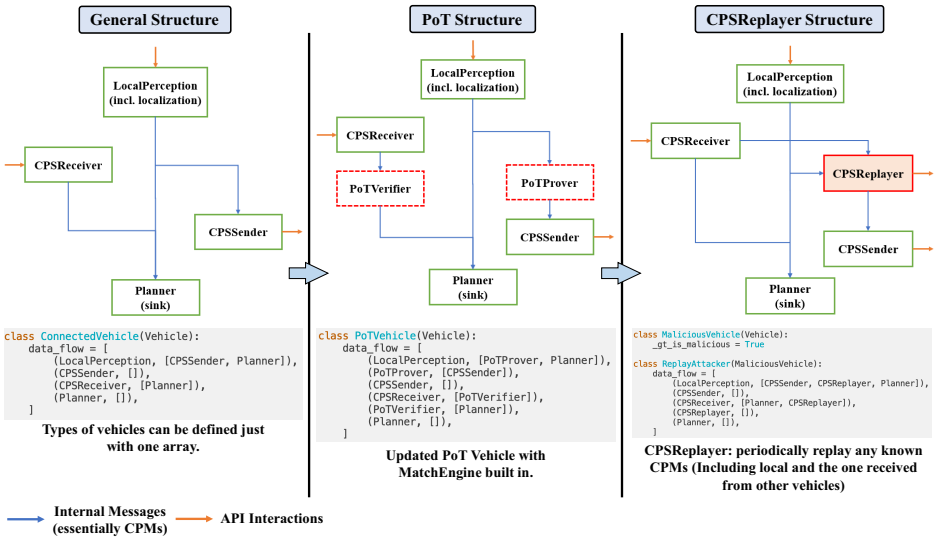


Figure 3.2: Example of defining vehicle types

To ensure the legitimacy of vehicle implementations, Flowsim incorporates a sandbox environment that explicitly controls the information accessible to agents and the operations they can perform. Various types of vehicles have been implemented in the sandbox. While the current focus is on vehicles, Flowsim can easily accommodate other types of agents with V2X capabilities, such as roadside units (RSUs), bicycles, and pedestrians with smartphones.

Currently, Flowsim supports several types of vehicles, as listed in Table 3.1. These include general-purpose vehicles like traditional unconnected autonomous vehicles, V2X connected vehicles, and the Proof-of-Traffic (PoT) vehicles [TJL+23]. Since cybersecurity is a key concern in the PoT context, Flowsim also implements different types of attacker vehicles with "tampered"

Table 3.1: Implemented vehicle types in Flowsim

Name	Description
UnconnectedVehicle	Traditional vehicles with perception, but not connected to V2X
ConnectedVehicle	Perception and V2X communication capable vehicles, can send and receive CPMs
PoTVehicle	Proof of Traffic capable vehicles, can send and verify proofs
SpamAttacker	Attacker vehicles sending random fake objects to make the road look congested
ReplayAttacker	Attackers replaying CPM of both received and local perceived objects to confuse or overload other vehicles
SilenceAttacker	Selfish vehicles that only listen to V2V communications without contributing anything

On-Board Unit (OBU) implementations, including spammers, replay attackers, and selfish attackers.

Flowsim also allows for the easy implementation of other vehicle types. For example, a human-driven vehicle can be represented as a dummy vehicle, which is equivalent to an empty OBU. By providing this range of vehicle types within a sandbox environment, Flowsim offers flexibility for simulating various scenarios and studying different aspects of cooperative autonomous systems.

3.1.3 Flow Inside the Vehicles

The design inside each vehicle follows a flow-based approach, which inspired the name **Flowsim**. The vehicle functions are abstracted into atomic and autonomous modules. Each module takes its input data, processes them, and produces the output data. Additionally, the modules can interact with the simulators outside of the sandbox by calling the explicitly provided APIs.

The data flow among the vehicle modules is represented by a Directed Acyclic Graph (DAG). In contrast to the widely-used Pub-Sub design pattern, we explicitly define and implement the data flow, using the adjacency list representation of the DAG. This explicit representation offers several advantages, including higher performance, easier design and debugging, and the elimination of the risk of infinite loops.

Furthermore, the data passing through the DAG are isomorphic, which means that all data transfers within a vehicle essentially consist of Collective Perception Messages (CPMs) [ETS23b] with private extensions. If required, some modules can call NetworkSimulator to broadcast those CPMs onto the network, after cleaning up the internally used private extensions.

Leveraging the strength and flexibility of Python, we developed a Domain Specific Language (DSL) that represents the adjacency list and enables the definition of different vehicle types using this DSL. As depicted in Figure 3.2, defining a vehicle based on a specific data flow is as simple as listing the module names and specifying the next modules to which the output of each module should flow. This flexibility enables us to manipulate vehicle types easily, including creating various types of attackers. By adopting a flow-based design approach, Flowsim provides a powerful and customizable framework for modeling and analyzing the behavior and interactions of vehicles within a simulation environment.

3.1.4 Metric Collection

Depending on the requirements, Flowsim enables the collection of various metrics after each tick of the simulation. The `MetricCollector` module serves as the utility for metric collection. Researchers can customize their implementation to produce metrics in formats that match their requirements, such as text, CSV, JSON, or DataFrame. In the current implementation, the `MetricCollector` serializes the metrics of each tick into separate lines of JSON. Moreover, it generates an "index" file that records the position of each line in the original file for rapid seeking.

Currently, Flowsim records four sets of metrics, three of which are generally applicable.

bytes_sent tracks the number of bytes sent by each vehicle in each tick, allowing for the calculation of bandwidth consumption.

local/received/all_objects provide insights into the accumulated number of objects collected by each vehicle from different sources. These include the objects perceived by sensors, objects received from cooperative perception, and the total number of objects available to the planner. These statistics are crucial for evaluating the performances and efficiencies of CAVs.

time_to_verify_histogram measures the distribution of the delay from receiving a new object to "believing" the data associated with it. This is specifically relevant to security-related research and can be used to compare the timeliness of different methods.

By collecting these metrics, Flowsim enables researchers to gain valuable insights into various aspects of the simulation, ranging from network performance to the behavior and efficiency of CAVs, as well as the timeliness of security-related processes.

3.2 Implementation Details

In this section, we will discuss the crucial implementation details that can impact the performance of Flowsim and our corresponding solutions.

3.2.1 Programming Language and Its Implementation

Python was selected as the programming language for Flowsim because of widespread use among researchers and specialties for rapid prototyping. To

maintain maximum readability, Flowsim was implemented in pure Python. Given the flexibility requirement and modular design, it was infeasible to write most parts in C extensions. Therefore, we selected the PyPy implementation.

PyPy is a Just-In-Time (JIT)-enabled Python implementation that offers several times faster execution than the official CPython implementation in most cases. It achieves all this under a hood, maintains full compatibility with the official CPython, and does not require modifications to the Python code. However, PyPy has some limitations, particularly when invoking C extensions. Calling C code interrupts the JIT compilation and significantly hampers performance, making it even worse than the CPython interpreter. This limitation is the main drawback of PyPy; therefore, it is crucial to minimize the time spent calling C code, which includes libraries such as `numpy` and interactions with `SUMO`.

It is important to note that despite its advantages, PyPy is not a solution to overcome the limitations imposed by the Global Interpreter Lock (GIL). The GIL still affects the performance of multi-threading in PyPy, rendering it inefficient for true parallel execution. To address the limitations imposed by the GIL, we have considered the possibility of employing the multiprocessing approach in Flowsim. It involves spawning multiple processes and facilitating communication among them to realize parallel execution. Although we have not yet implemented multiprocessing in Flowsim, it is a potential solution for enhancing performance.

Switching to multiprocessing typically requires a significant program redesign. However, the sandbox design of Flowsim isolates the vehicles and defines explicit APIs between vehicles and simulators; thereby minimizing the need for extensive modifications. If we were to implement multiprocessing, we would spawn worker processes for vehicle agents and utilize Remote Procedure Calls (RPCs) to wrap API calls to the simulators. This approach enables the efficient utilization of system resources and can boost the overall performance of Flowsim.

By leveraging the PyPy implementation and considering the potential use of multiprocessing, Flowsim strikes a balance between performance optimization and code maintainability, enabling researchers to perform simulations efficiently while enjoying the readability of Python.

3.2.2 PositionManagerV2

PositionManagerV2 is the current implementation of the Position Manager in Flowsim. It is responsible for synchronizing the list of vehicles and their position information with the external traffic provider SUMO. One of the key

functions of the Position Manager is `get_nearby_vehicles()`. The function checks all the vehicles on the map and returns those that are within a predefined distance from the egovehicle. In the naïve implementation, iterating over all vehicles results in a time complexity of $O(N)$, where N indicates the total number of vehicles.

To optimize this process, we introduce an algorithm that divides the map into orthogonal grids of a predefined size. Each vehicle is then assigned to a specific grid cell according to its current position. This approach allowed us to answer each query efficiently by filtering through the egovehicle's cell and its eight adjacent cells. Using this optimization, we achieved a time complexity of $O(n)$ per query, where n represents the number of vehicles in each cell. The preprocessing step requires $O(N)$ time to assign vehicles to the grid cells.

To handle corner cases, such as when the coordinate becomes -1.6 because the centerline of the edge road is 0, we surround the map grids with margins of two cells on each side. Although using hexagonal grids could offer slightly better efficiency, we opted for orthogonal grids for the sake of simplicity and better readability of the implementation.

By employing this optimized algorithm for querying nearby vehicles, PositionManagerV2 significantly improves the performance of Flowsim, enabling efficient and accurate retrieval of vehicles within a specified distance from the egovehicle.

3.2.3 PerceptionSimulatorV3

PerceptionSimulatorV3 is the current implementation of the perception simulator in Flowsim. In this version, we utilized a simple computer vision model to simulate perception capabilities. The key features of this model are as follows:

Two-dimensional world The perception simulator assumes a world with only two dimensions, disregarding the altitude or Z-axis.

Field of view (FOV) The field of view of the camera is thus reduced to a one-dimensional representation, presented as an angle range.

Simplified vehicle shapes All vehicles in the simulation are represented as perfect rectangles to facilitate easier geometric calculations.

Numberplate placement Numberplates are simulated as shorter lines located at the center of the front and rear bumpers of the vehicles, thus they are centrosymmetric about the center of the vehicle.

Exclusion of polygons This implementation does not consider polygons such as buildings in the simulation.

Although we have simplified the perception algorithm to be purely geometric-based, it remains the most computationally intensive task within the simulator. The perception algorithm in PerceptionSimulatorV3 can be implemented in the following five steps:

Vehicle Reconstruction

In Flowsim, we reconstructed the dimensions of the vehicles based on the position information provided by SUMO. We accurately determine the dimensions and shape of the vehicles in the simulation by utilizing the position of the center of the front bumper and the heading of each vehicle. The reconstruction process ensures realistic representations of the vehicles and enables accurate interactions and behaviors within the simulated environment.

Coordinate conversion and FOV filtering

The next step involves converting the retrieved vehicle boxes from the original "world" coordinate system into "camera", a Cartesian coordinate system originating from the front camera of the egovehicle. Specifically, the camera of the egovehicle is located at $(0, 0)$ and is directed towards the positive x-axis.

Let us consider the original "world" coordinate system with the egovehicle's camera located at (x_0, y_0) with a heading of β_0. If a vehicle is located at (x, y) with a heading of β in this coordinate system, it can be projected onto the new coordinate system $[x', y', \beta']$ using Equation 3.1.

$$\begin{bmatrix} x' \\ y' \\ \beta' \end{bmatrix} = \begin{bmatrix} \cos(\beta_0) & \sin(\beta_0) & 0 \\ -\sin(\beta_0) & \cos(\beta_0) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - x_0 \\ y - y_0 \\ \beta - \beta_0 \end{bmatrix} \tag{3.1}$$

Once the vehicles are converted into a new coordinate system, it becomes simple and straightforward to filter out the irrelevant vehicles based on the camera's FOV. In Figure 3.3, we can observe that a vehicle is considered irrelevant only if NONE of its corners, denoted as A, B, C, D, fall within the FOV. In other words, these vehicles have no chance of being identified or of occluding other vehicles.

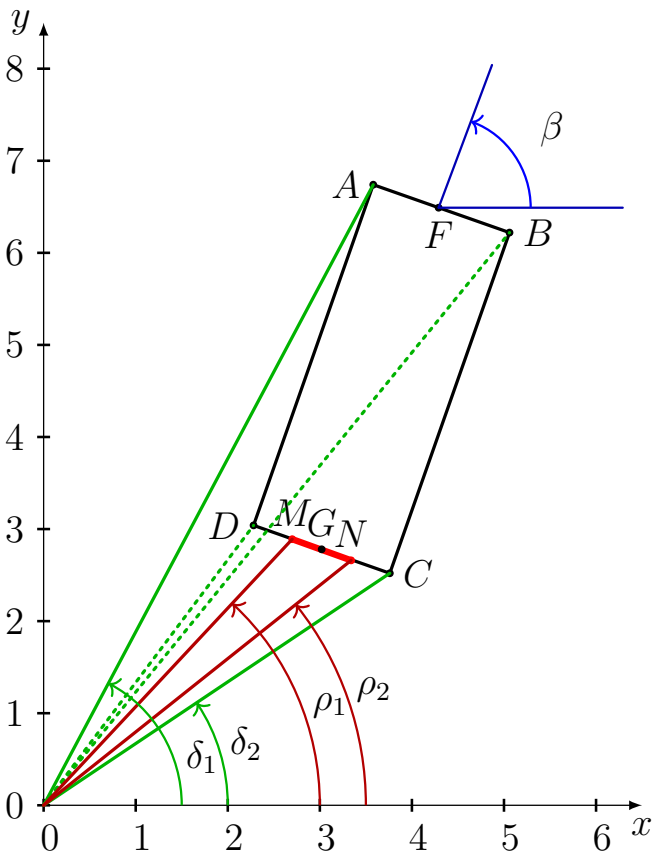


Figure 3.3: Relevant points and corresponding angles in the camera coordinate.
Green lines: projections of corners of vehicles;
Red lines: projections of the rear numberplate;
β: heading angle related to the camera position;
δ_1, δ_2: argument angles of corners of vehicles;
ρ_1, ρ_2: argument angles of the rear numberplate.

Calculate the Projection

The centrosymmetric assumption of the vehicles and their numberplates allowed us to normalize the "heading" direction of the vehicle with respect to the camera. Specifically, if $|\beta|$ (the absolute value of the heading) is greater than $\pi/2$, the vehicles are rotated by π radians. This normalization ensures that the vehicles are always effectively "tailing" the camera, allowing us to focus solely on their (equivalent) rear numberplates.

Referring to Figure 3.3, we can narrow our focus to six key points: corners A, B, C, D, and the ends of the numberplates, denoted as M and N.

Since the FOV of the camera is represented by angles, we can simplify the calculations by transforming the coordinates once again into polar coordinates. However, for efficiency purposes, we do not need a complete polar coordinate representation of the points. Instead, we only require the arguments (i.e., the angle from the positive x-axis to the point).

Thus, we can calculate the range of the vehicle box (δ_1, δ_2) and the range of its numberplate (ρ_1, ρ_2) using Equation 3.2.

$$\begin{aligned} \delta_1 &= \min(Arg(A), Arg(B), Arg(C), Arg(D)) \\ \delta_2 &= \max(Arg(A), Arg(B), Arg(C), Arg(D)) \\ \rho_1 &= \min(Arg(M), Arg(N)) \\ \rho_2 &= \max(Arg(M), Arg(N)) \end{aligned} \tag{3.2}$$

$Arg(P)$ gives the argument of a point P ranging from $-\pi$ to π, where min and max denote getting the minimum and maximum value, respectively.

Subsequently, we sorted the candidate vehicles based on their distance from the center of the rear bumper, denoted as G.

Occlusion Calculation

Given that the current implementation ignores the polygons from the map, the only factor to consider is the occlusion between two vehicles.

To facilitate occlusion calculation, we mapped the argument angles δ_1, δ_2, ρ_1, ρ_2 within the polar coordinate system onto a "flattened" number line ranging from $-\pi$ to π. This mapping allows us to transform the problem of "occlusion" into an interval coverage problem.

The algorithm for this step is defined as Algorithm 1. By leveraging the segment tree [dB08] data structure and employing the discretization method, the time complexity can be reduced from $O(|C|^2)$ to $O(|C| \log |C|)$, where $|C|$ represents the number of candidate vehicles.

Algorithm 1 get_visible_lines

Require: C as candidate vehicles in projection format, sorted by the distance of G
Ensure: Visible vehicles in projection format

```
if C = ∅ then
    return ∅
S ← ∅
R ← [C^0]
for (ℓ, m) in zip(C[: −1], C[1 :]) do
    S ← S ∪ {δ_1^ℓ, δ_2^ℓ}
    if (ρ_1^m, ρ_2^m) ∉ S then
        R ← R ∪ [m]
return R
```

Filter by Heading

Finally, even if the numberplate of a vehicle is not occluded, it may still be considered unrecognizable if it is rotated too much. Therefore, we filtered the candidate vehicles again according to their headings, which is β in the camera coordinate system. This filter ensures that only vehicles within an acceptable rotation range are considered identifiable.

By going through these steps, we can ensure that the remaining vehicles can be perceived and identified by the camera of the egovehicle. The PerceptionSimulatorV3 records the ground truth and returns the corresponding numberplates to the egovehicle.

3.3 Verification

In this section, we verify the functionality and performance of Flowsim based on specific settings. It is important to note that Flowsim was initially developed to evaluate the work Zero-Knowledge Proof of Traffic (zkPoT) [TJL+23]. As a result, certain implementations and experiment parameters were tailored for this scenario. For the experiments, we selected a 24-hour realistic traffic dataset generated from the road network of Luxembourg, known as the LuST scenario [CFE15]. The LuST scenario includes 280,000 distinct vehicles that have been part of the scenario at some point. During the peak hour at 19:00, there are over 8,000 active vehicle agents to be simulated. To maintain a satisfactory level of simulation accuracy, the tick resolution was set to 1 second per tick. Consequently, the full-sized LuST scenario can be completed in less than one day of wall time.

Throughout the experiments, we collected various metrics per tick per vehicle, including the positions of the vehicles, the number of bytes sent, and the number of known objects from different modules. This comprehensive data collection approach provides maximum flexibility for data analysis, enabling us to examine the required information and generate relevant visualizations. Figure 3.4 shows examples of the data aggregated from the collected metrics.

Figure 3.4a shows a line chart depicting the average bandwidth consumption over the simulation time. The data enables us to assess the impact of the proposed method on the network. It was obtained by calculating the mean value of all the vehicles for each tick.

Figure 3.4b presents a pie chart representing the Time-to-Verify (TTV) for a given tick. TTV indicates the number of seconds of delay before the newly received vehicle data are protected by cross-verification. This metric measures the timeliness of the proposed method and can be compared with conventional statistic-based approaches. It was obtained by summing up the TTV histogram data for each vehicle activated by the tick.

Figure 3.4c shows a heatmap of Cooperative Perception Ratio (CPR) of our work at 19:00. CPR reflects the contribution level of information received from other vehicles compared with information gathered from local perceptions. The metric is calculated by dividing the sum of the objects not perceived locally by the sum of the locally perceived objects.

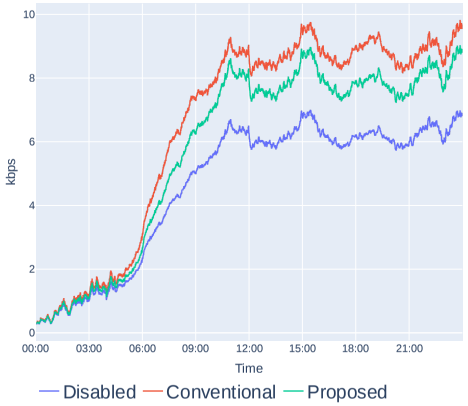
The experiments were conducted on a desktop PC equipped with an Intel i9-13900kf CPU and 128GB DRAM.

We timed the common scenario, in which all vehicles were capable of cooperative perception. Figure 3.5 illustrates the total processing time per tick and provides a breakdown of the simulators and agents. A maximum processing time of 1.4 seconds per tick was observed when the number of active vehicles reached 8,200. Most ticks were processed in approximately 600 milliseconds, which indicates an average processing time of approximately 15 microseconds per vehicle.

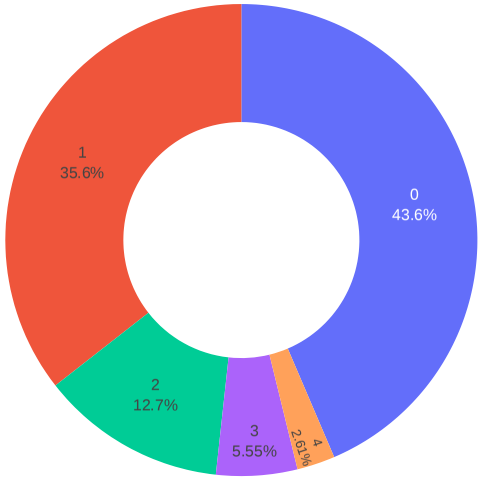
3.4 Application

Thanks to its extensible and flexible nature, Flowsim exhibits the capability to be modified and extended, thereby accommodating a diverse range of use cases. These use cases encompass various aspects, as outlined below:

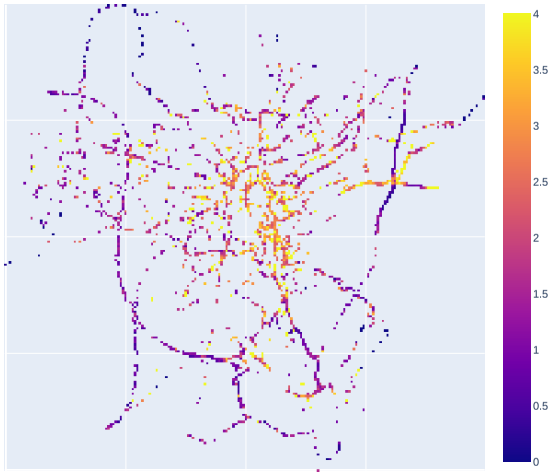
Behavioral Analysis Flowsim enables the study and analysis of the behavior of CAVs, providing insights into their interactions, driving policies, and overall behavior in different traffic conditions.



(a) Line: average bandwidth consumption



(b) Pie: Time-to-Verify (TTV) distribution



(c) Heatmap: cooperative perception ratio

Figure 3.4: Example figure types generated from collected metrics

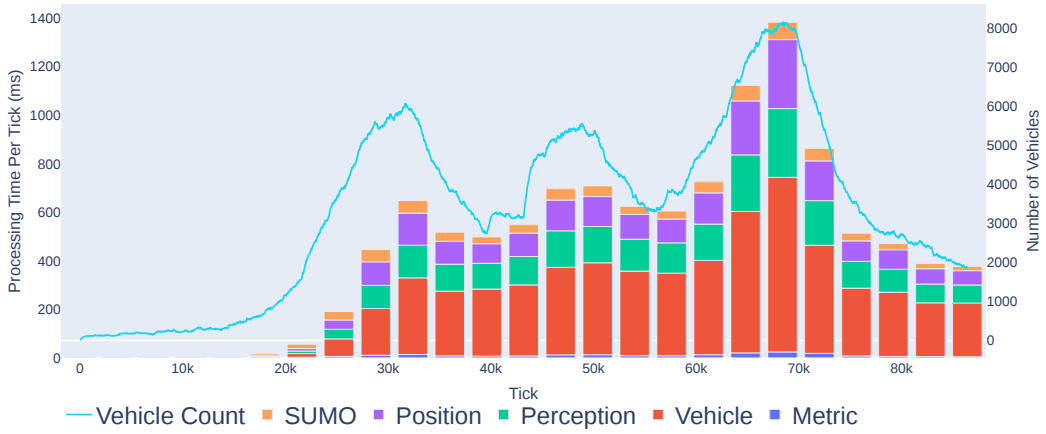


Figure 3.5: Number of vehicles and processing time breakdown

Protocol Evaluation Flowsim facilitates the evaluation of communication and networking protocols for CAVs, assessing their performance, safety, and reliability in CAV networks.

Cybersecurity Assessment Flowsim supports the assessment of cybersecurity vulnerabilities in CAVs, enabling researchers to simulate attack scenarios, evaluate security measures, and develop robust countermeasures.

Traffic Optimization Flowsim provides a platform for evaluating and optimizing traffic management strategies, allowing researchers to test traffic control algorithms, congestion management, and route planning techniques for improving traffic flow and reducing travel times.

Policy Development Flowsim aids in the development and evaluation of policies and regulations for CAVs, enabling simulation-based assessments of various interventions, such as lane assignment, speed limits, and priority rules on traffic flow, safety, and energy efficiency.

3.5 Summary

In this study, we presented Flowsim, a novel simulator designed to evaluate the behavior and data flow of CAVs. Recognizing the limitations of existing simulators in capturing the intricacies of CAV behaviors, Flowsim was developed with a focus on modularity, extensibility, and performance to address the unique requirements of behavior evaluation. We introduced the motivation behind Flowsim, highlighting the need for a simulator that specifically

targets the behavior of CAVs. Existing simulators are useful for various aspects of transportation and network simulation, but they often fail to provide a comprehensive evaluation platform for CAV behaviors. Flowsim fills this gap by offering a city-scale experimental environment, high-performance simulation capabilities, and a modular architecture that can be customized and extended to suit specific research needs.

Through carefully designed experiments, we assessed the functionality and performance of Flowsim. These experiments ensure the ability of Flowsim to capture realistic CAV behaviors, evaluate the effectiveness of cybersecurity measures, and assess the impact factors on CAV performance.

Flowsim is written in merely 1500 lines of pure Python code, making it highly readable, understandable, and easily modifiable for different applications. We not only welcome researchers to use Flowsim for their studies but also strive to create a learnable and playable open environment. By providing accessible and well-documented code, we aim to foster collaboration and knowledge sharing within the research community. The code of Flowsim is available in the largest opensource community GitHub. Any contributions and pull requests are highly welcome.

In future work, we aim to further enhance Flowsim by integrating more advanced simulation components, such as advanced perception models and detailed communication protocols, and implementing planning and control algorithms. Moreover, we plan to explore the application of Flowsim in various research fields, including traffic optimization, intelligent transportation systems, and cooperative driving scenarios. By continuously improving and expanding Flowsim's capabilities, we endeavor to contribute to the advancement of CAVs and foster safer and more efficient transportation systems.

Chapter 4

Zero-knowledge Proof of Traffic

Cooperative perception is a crucial component of C-ITS. However, ensuring the authenticity of remote perception data remains a challenge, as the vehicles cannot verify events that they do not witness independently.

In this chapter, we will propose a novel cryptographic approach called zero-knowledge Proof of Traffic (zk-PoT), which involves generating proofs to the traffic observations, which can then be deterministically cross-verified by any receivers without relying on ground truth, probabilistic, or plausibility evaluations. Additionally, no private information is compromised during the entire procedure. The proposed approach can be incorporated into existing ETSI and ISO standards, without disrupting the backward compatibility. [1]

4.1 Problem Statement, Assumptions and Approaches

4.1.1 Problem Statement

The cooperative perception ability provided by CPMs is significant; however, the potential risks associated with such messages must be considered. Blindly trusting CPMs could lead to suboptimal or even risky vehicle decisions, which can considerably compromise road safety.

[1] Contents of this chapter have been published as the following articles: **Ye Tao**, Ehsan Javanmardi, Pengfei Lin, Yuze Jiang, Jin Nakazato, Manabu Tsukada, Hiroshi Esaki. "Zero-Knowledge Proof of Traffic: A Deterministic and Privacy-Preserving Cross Verification Scheme for Cooperative Perception Data." IEEE Access. And, **Ye Tao**, Yuze Jiang, Pengfei Lin, Manabu Tsukada, Hiroshi Esaki. "Zk-PoT: Zero-Knowledge Proof of Traffic for Privacy Enabled Cooperative Perception." 2023 IEEE 20th Consumer Communications & Networking Conference (IEEE CCNC 2023).

Some studies [MZTC11] are based on trust estimation and management methods that depend on past statistics, making it nearly impossible to detect one-shot misinformation. Other studies rely on assessing the "plausibility" of data, which also necessitates the use of past statistics. Furthermore, these models cannot deterministically verify data as they are statistical models.

Therefore, we designed a cryptography-based mechanism that allows vehicles to prove their traffic observations, enabling other vehicles to cross-verify and *deterministically* trust the observations without knowing the ground truth.

Our study is primarily focused on proving the existence of observed vehicles as "existence" is the most fundamental and accurate event that occurs on the road, which remains unchanged despite the high speed of vehicles. This is contrary to a vehicle's location, which can change quickly, leading to inaccuracies. Although fundamental, proving existence can already block many data-fabricating attacks, such as the most widely discussed phantom vehicle attack.

4.1.2 Assumptions

In this study, we make several assumptions. Firstly, We assume that all vehicles have a front camera system and can recognize the other vehicles' number plates by computer vision. We also assume that every vehicle on the road joins VANET, enabling all vehicles to perceive and share information with other vehicles. Consequently, we do not differentiate between conventional vehicles vs. vehicles with cameras, connected vs. non-connected vehicles, in our subsequent discussions.

Additionally, we assume that a public key infrastructure (PKI) run by LEAs is deployed to all the infrastructures and vehicles. Furthermore, vehicle certificates with key pairs are distributed to all the vehicles by LEAs; each vehicle signs every message it sends and uses the public key to verify the integrity and authority of the messages they receive. We also assume that all the vehicle certificates are provided by a pseudonym system with perfect unlinkability. This implies that after a vehicle changes its pseudonym, i.e. its pseudonymous certificate, the new pseudonym cannot be linked with the old one.

Based on these assumptions, we define "A heard B" as A receiving a message that contains B's pseudonym from V2X communication. Conversely, we define "A saw B" as A identifying B's number plate, extracted from the video feed of its front camera.

4.1.3 Approaches

We can design a naïve "proof system", which lets vehicles broadcast the value of vehicle number plates that they observe. However, such a "proof system" is not secured as the plate numbers can be easily synthesized, which makes it subject to data forging, replay attacks, and other threats. Moreover, the location privacy of the observed vehicles is compromised, since the number plates are in plaintext and can be used to track vehicles by malicious parties.

To address these issues, we propose a zero-knowledge proof (ZKP) based system that enables vehicles to prove that they have observed another vehicle. Based on the definition of "seeing" and "hearing," we consider that the vehicle that can link the number plate and pseudonym must have a close observation of the target vehicle, presenting strong evidence of the existence of the target. Additionally, our ZKP-based system ensures that both the pseudonyms and number plates remain undisclosed in the proofs, thus ensuring location privacy.

In our system, it is difficult if not impossible to falsify the existence of a random vehicle. To overcome this limitation, we propose a novel approach that employs a cross-verification scheme. Our approach involves multiple vehicles generating individual proofs for the same target vehicle. Thus, any third party can verify the existence of the target vehicle by comparing multiple proofs without requiring any information regarding the specific vehicle.

4.2 Zero-Knowledge Proof of Traffic

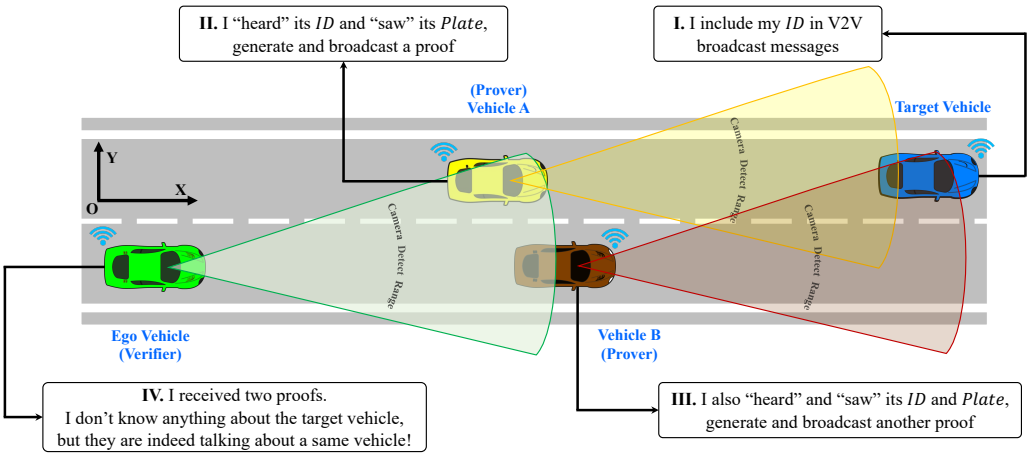


Figure 4.1: Procedure of a Zero-Knowledge Proof of Traffic

In this section, we present our proposed solution, Zero-Knowledge Proof of Traffic (zk-PoT), to the problem of verifying whether two vehicles have observed the same target vehicle while preserving their location privacy. We convert the problem into a cryptographic protocol called zero-knowledge proof of shared secret (zk-PoSS). We then provide a solution to this model and apply it to the existing ETSI standard by extending the packet structure and station behaviors. We use ETSI standards as an example; our approach can also be applied to other cooperative perception systems, such as ISO standards.

4.2.1 Problem Conversion

Fig. 4.1 depicts a scenario where vehicles A and B observe a vehicle named Target and acquire its pseudonym, ID, and number plate, $Plate$. These observations essentially yield a *shared secret*, which is derived from the target vehicle's ID and $Plate$. To prove that they indeed know the shared secret, without disclosing the plaintext ID and $Plate$, we proposed a cryptographic protocol called zk-PoSS. This model involves creating two distinct proofs using a one-way proof function, F_{proof}, using a shared secret, SS, and two cryptography salts, m and m'. Subsequently, another function, F_{verify}, is used to determine if the pairs, (P, m), and (P', m'), are correspond to the same shared secret, SS.

By doing so, the problem of proving a common observation to a vehicle could be transferred to the cryptographic model of proving the shared secret:

$$\begin{aligned} &P = F_{proof}(SS, m) \\ &P' = F_{proof}(SS, m') \\ &P \neq P' \\ &F_{verify}(P, m, P', m') = True \end{aligned} \tag{4.1}$$

4.2.2 Zero-Knowledge Proof of Shared Secret

After converting the practical problem of zk-PoT into the theoretical problem zk-PoSS, it can be solved using the conventional ECDSA.

The conceptual overview of zk-PoSS is shown in Fig. 4.2. Concatenating the ID and plate of a specific observed vehicle, the provers essentially possess the same shared secret with sufficient entropy. Since the private key of ECDSA is just a plain integer, they can interpret the hash of such a shared secret as a private key. Therefore, two valid but different signatures become

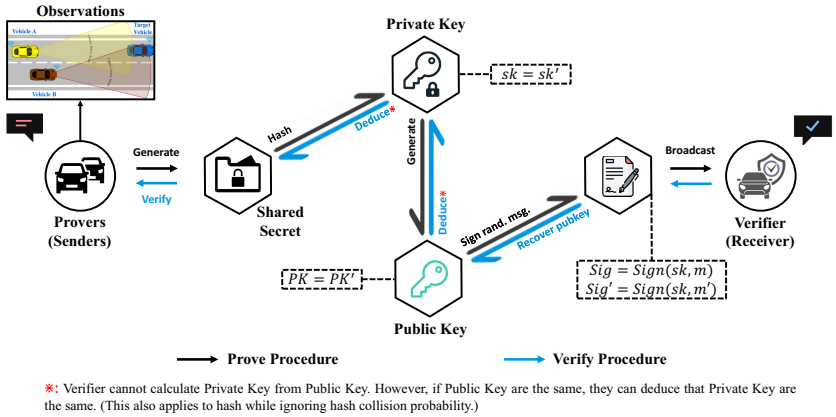


Figure 4.2: Overview of Zero-knowledge Proof System

proof that the provers possess the same private key. In the proposed scheme, holding the private key implies knowledge of the shared secret.

Utilizing this feature of digital signature, two parties can create different proofs regarding the shared secret, by signing different random messages as salts. Subsequently, a third party that received both proofs can recover the public keys from the signatures, and verify if the two public keys are identical. If these public keys are identical, it can deduce that the private keys are identical; and that the private key is, in essence, a hash of the shared secret. Therefore, from the proof by contradiction, the only feasible option is that both indeed share the same secret.

The exact protocol is defined in Protocol **zk-PoSS**.

Protocol **zk-PoSS**

Preconditions: Common Agreement

Prior to initiating the protocol, there is a set of common parameters that all parties must agree upon. This ensures that all parties are working with the same fundamental assumptions, enabling coherent and meaningful communication.

1. All parties agree with the same set of ECDSA parameters:

$$\{CURVE, G, n\}$$

   These parameters define the elliptic curve, the base point, and the order of the base point, respectively.

2. All parties adopt the same cryptographic hash function, denoted by H. This function is capable of producing a fixed-length output that matches the binary length of n. This uniformity ensures consistency across all hashed outputs.

Prover: Proof Construction

The prover's role is to generate a proof of knowledge to a secret, without revealing the secret itself. The following steps outline this process.

1. The prover computes the hash of the secret SS using the agreed cryptographic hash function:

$$sk = H(SS)$$

   Here, the hashed secret, sk, could serve as a private key in the ECDSA context.

2. There may be rare cases where $sk >= n$ since in ECDSA n is a prime which is slightly less than a power of two. To address these outliers, we perform an iterative process of recalculating the hash by repeating SS, until $sk < n$.

3. The prover then computes

$$PK = G \times sk$$

   on curve $CURVE$. The generated pair (sk, PK) can be interpreted as the private key and public key of the ECDSA algorithm, respectively.

4. Using the private key sk, the prover signs a random message m with the function:
$$Sig = ECDSA_Sign(sk, m)$$
This signature attests to the ownership of sk.

5. The prover publishes the message m and its corresponding signature Sig, available for any verifier to check.

Verifier: Pairing and Verifying Proofs

The verifier's role is to match and validate the proofs provided by the provers, thus affirming the provers' common knowledge of the shared secret. Here are the steps a verifier must follow.

1. The verifier recovers the public key PK from the message m and signature Sig using the function:
$$PK = ECDSA_Recover(m, Sig)$$
The recovered public key is then stored in a database locally.

2. If the database already contains another message m' with corrsponding valid signature Sig' which is signed using the same public key PK, it implies that both provers of m and m' possessing such PK share the same secret. This is a significant conclusion as it allows us to infer shared knowledge without revealing the secret itself.

A zero-knowledge proof system must satisfy three properties [GMR89]:

1. *Completeness*: If the statement is true, the verifier can be convinced by the prover.

2. *Soundness*: If the statement is false, the verifier cannot be convinced by the prover, even if the prover tries to cheat.

3. *Zero-knowledge*: The verifier cannot gain any knowledge apart from the truth of the statement.

Zk-PoSS is considered a zero-knowledge proof system because it satisfies all three properties.

1. *Completeness*: If PKs are identical, the verifier could deduce that the SS used to generate the PKs are identical, although it cannot know the actual secret.

2. *Soundness*: If SS are not identical, then the same sk can only be produced through a hash collision, which is a computationally difficult task. Thus, a cheating prover has a negligible chance of convincing the verifier.

3. *Zero-Knowledge*: In the whole process, the verifier cannot produce any information regarding sk or SS from PKs. This property is guaranteed by any public key cryptography, including ECDSA.

4.2.3 Integrating zk-PoT into the ETSI CPS standards

The ETSI CPS system can be updated by incorporating the zk-PoSS protocol to provide proof of traffic capability. The standard curve `secp256k1` is used with a 256-bit key length, which will produce a 65-byte long signature. The message, m, used in zk-PoSS is the prover's pseudonym, which is selected for its high entropy and frequent transmission in every packet, enabling bandwidth conservation. Furthermore, it helps in preventing the naïve replay attack, which will be discussed in Section 4.4.

Changes to the CPM packet structure

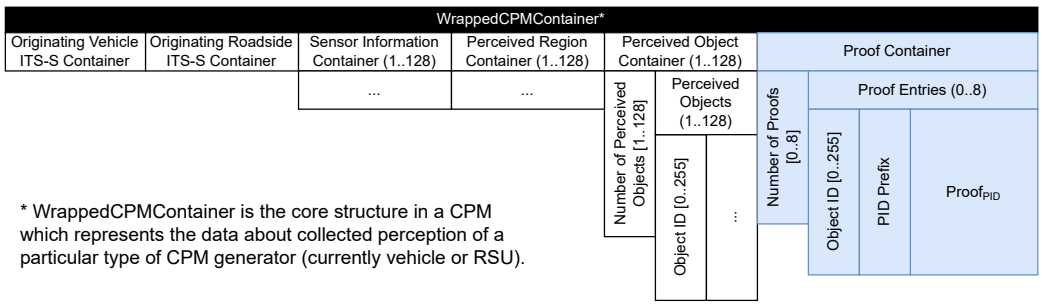


Figure 4.3: Structure of the extended part of collective perception message (CPM) for zk-PoT

The `WrappedCPMContainer` structure in the CPM has been extended with an additional section called the `Proof Container`, which is located at the end, contains a list of 0 to 8 `Proof Entries`, as shown in Fig. 4.3. Each `Proof Entry` contains an `object ID` that is present in the perceived object container, a 32-bit prefix of the vehicle's ID to assist quick filtering, along with the actual proof of this ID.

The ASN.1 definition of `ProofEntry` is as follows.

```
ProofEntry ::= SEQUENCE{
    objectID Identifier2B,
    pidPrefix Integer32,
    v BOOLEAN,
    r OCTET STRING (SIZE(32)),
    s OCTET STRING (SIZE(32))
}
```

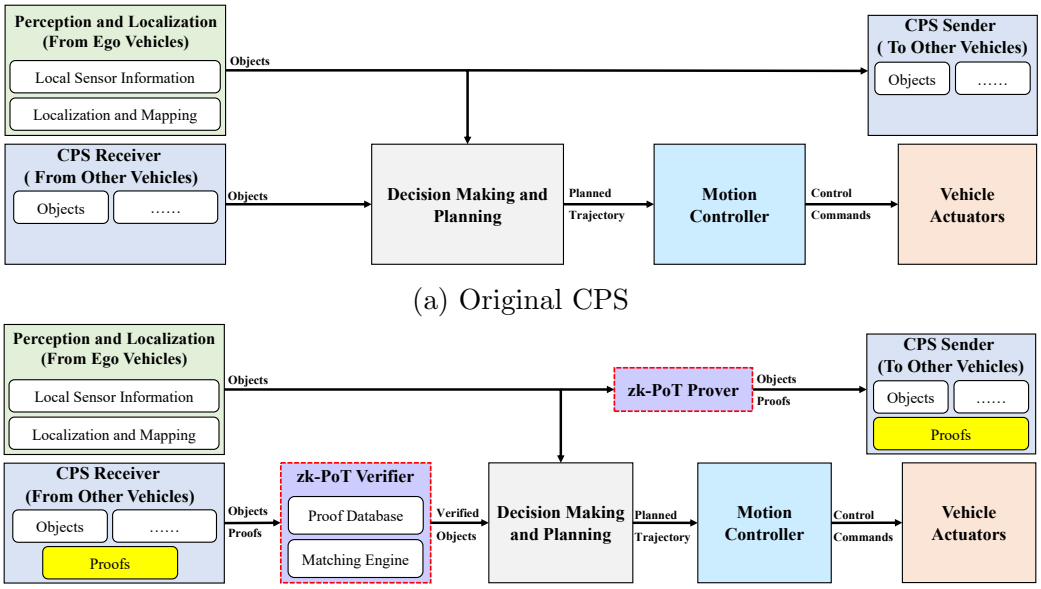
The `objectID` field represents the ID of the target vehicle assigned in `PerceivedObjects`, the `pidPrefix` field represents the 32-bit prefix of the target vehicle's pseudonym, and the three fields `V`, `R`, and `S` represent the corresponding fields in the ECDSA signatures, as defined in [Bro09].

It is recommended that the proofs be sent intermittently rather than in every CPM to minimize the amount of data transmitted and preserve bandwidth, As the CPM can be broadcasted at a high frequency of up to 10 Hz [ETS19], the length of the proofs (71 bytes per `proof entry` under ASN.1 UPER encoding) can add up quickly. Therefore, it is encouraged to implement an inclusion management process to limit the number of `proof entry` in the prover; and the receiver should store each entry for a specified period. The exact time interval for transmitting the proofs is not yet determined; however, sending them once every 3 s is considered sufficient to handle vehicle topology changes.

Changes to the ITS Station

Various implementations may produce different module designs; however, the fundamental structure of a vehicle equipped with the CPS should be similar to the one depicted in Fig. 4.4a. The `receiver` module retrieves objects from the received CPMs, and in conjunction with locally perceived objects, feeds them into the `planning` module for further processing. Additionally, the objects obtained from the `local` module are sent into the `sender` module to be packed as CPMs and broadcasted to the vehicles nearby.

To accommodate the zk-PoT, we introduced two new modules: `prover` and `verifier`. The `prover` module obtains objects from `local`, creates proof for each potential object, and then transfers the proofs along with the original objects to the `sender` module for CPM preparation and transmission, as shown in Fig. 4.4b. On the receiver side, the `verifier` module, positioned after the `receiver`, functions as a gatekeeper, ensuring that only verified received objects can be used for the `planning` module. It receives raw objects and proofs, stores them in an internal database, and continually verifies these proofs. When a match is found, the corresponding objects are considered



(a) Original CPS

(b) Extended CPS

Figure 4.4: Block Diagram of Original v.s. Extended CPS

authentic, and all the recent data of this object, including that stashed in the internal database, are passed to the `planning` module. Additionally, any subsequent data from the same objects are immediately forwarded to the `planning` module.

Such design ensures minimal modifications to the pre-existing modules while incorporating the proof of traffic functionality. Furthermore, this approach maintains backward compatibility with the current standards.

4.3 Evaluation

To evaluate the efficiency and performance of the zk-PoT compared to non-verified CPS in a large-scale and realistic environment, we proposed the Flowsim [TJN+23], a modular simulation platform for microscopic behavior analysis of city-scale connected autonomous vehicles. Various microscopic simulations were conducted on the Flowsim to evaluate the robustness, attack resistance, and network efficiency of zk-PoT.

The vehicles currently implemented are listed in Table. 3.1.

4.3.1 Parameter and Metric Settings

A set of four experiments is designed to compare the zk-PoT with the conventional methods under different conditions:

- Local Perception Only: all the vehicles operate standalone and do not communicate with each other.
- Conventional CPS: all the vehicles participate in the V2V communication and share their perception based on the ETSI CPS standards.
- Proof of Traffic (1s repeat): all the vehicles are capable of Proof of Traffic, which includes (up to 8) proofs for recently seen vehicles in the CPM they sent every second.
- Proof of Traffic (3s repeat): identical to PoT, except the vehicles only repeat proofs of the same vehicle once every three seconds. This setting can potentially decrease the overall bandwidth overhead while maintaining comparable performance.

These experiments are conducted in scenarios with different scales. Table. 4.1 lists the common parameters throughout all the experiments.

Table 4.1: Common Parameter Settings

Parameter	Value
Vehicle Length	4.0 meter
Vehicle Width	1.8 meter
Numberplate Width	0.35 meter
Perception Distance	65.0 meter
Expected pseudonym valid time	12 hours
Pseudonym changing probability	1/43200 per second
Camera Sensing Angle	120.0 degree
Communication Range	300 meter
Communication Delay	1 ms
Packet Delivery Ratio	80 %

We collect the following metrics from all the experiments:

Time to Verify (TTV) of a specific observed vehicle denotes the time delay between the first proof of this vehicle being received and successful cross-verification, which involves receiving the proof from another observer in our case. *TTV* conceptually represents the time required for an observed vehicle to be covered by cross-verification. This metric can facilitate the

design of the planning module. It can also evaluate the overall performance of the zk-PoT.

Local / Received / All Objects represent the number of accumulated objects collected in the different modules. Local Objects (N_l) is obtained from `LocalPerception`; Received Objects (N_r) is obtained from `CPSReceiver`; All Objects (N_a) is obtained from `Planner`. These three metrics can evaluate the performance of the CPS, as well as the impact of zk-PoT in the long term.

Verification Ratio (R_{veri}) is defined as the ratio of cross-verified objects (N_a) to the total received objects (N_r). They both excluded the local perceivable objects (N_l) because this portion does contribute additionally to the ego vehicles. It intuitively measures the percentage of the vehicles that could eventually be cross-verified over the long run.

$$R_{veri} = \frac{N_a - N_l}{N_r - N_l} \tag{4.2}$$

Bandwidth Consumption measures the bandwidth overhead produced by the proofs, and can also evaluate the effectiveness of the bandwidth conservation of PoT3.

4.3.2 Manhattan Scenario: Bootstrap and Convergence

Firstly, a set of small-scale experiments was conducted on a synthetic grid map, also known as the Manhattan scenario. This scenario primarily facilitates debugging and testing along with the development of Flowsim, but it can also be used to evaluate the bootstrapping and converging progress. The map contains 10 by 10 grids with the sizes of each grid set to 100×100 meters, as shown in Fig. 4.5 (vehicles scaled up for better visibility). Initially, 100 vehicles are spawned in the scenario, with random starting points and destinations. No vehicles are spawned or removed afterward during the whole experiment.

Each experiment was run for two hours (7200 ticks) and the number of vehicles available for `Planner` module is shown in Fig. 4.6. *Unconnected* presents the number of which no V2V communication is involved, equivalently local perception only, which converges the slowest since every vehicle needs to meet each other once. *Conventional CPS* converges the fastest since it does not require cross-verification. *Proof of Traffic (1s repeat)* converges slightly slower than the conventional non-verification approach. *Proof of Traffic (3s repeat)* converges negligibly slower than *1s repeat*, but it conserves a significant amount of bandwidth, which helps in mitigating the congestion when density increases. For comparison, the number of tracked vehicles

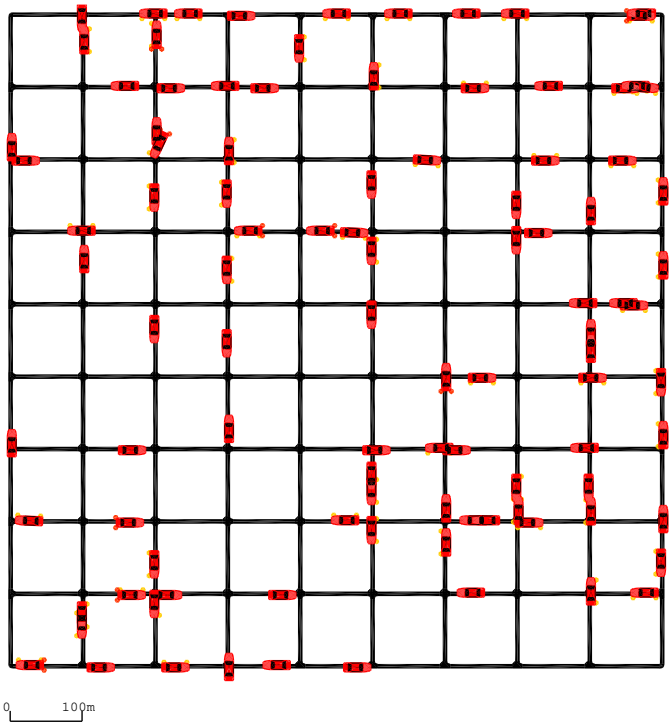


Figure 4.5: Manhattan scenario

reaches 95 percent in 14 minutes. Conversely, Proof of Traffic reaches the same point in 20 minutes. They both converge much faster than the unconnected setting, which takes 68 minutes. The results demonstrate that the Proof of Traffic has a relatively fast cold start speed in such a scenario.

4.3.3 LuST scenario: City-scaled Long-term Experiments

The final and most important group of experiments is the LuST scenario. We use the same scenario in this work as that of the previous work, which is the 24-hour realistic traffic data generated from Luxembourg (a.k.a. the LuST scenario) [CFE15].

As shown in Fig. 4.7, the road network contains diverged types of roads and intersections, which makes it suitable to analyze the effectiveness of zk-PoT in different situations. On the map, we can observe a U-shaped road stretching around the city. This is the European Route E44, which is also known as the A1 Motorway of Luxembourg, the road with the most dense traffic. However, in the LuST scenario, there is a problematic roundabout

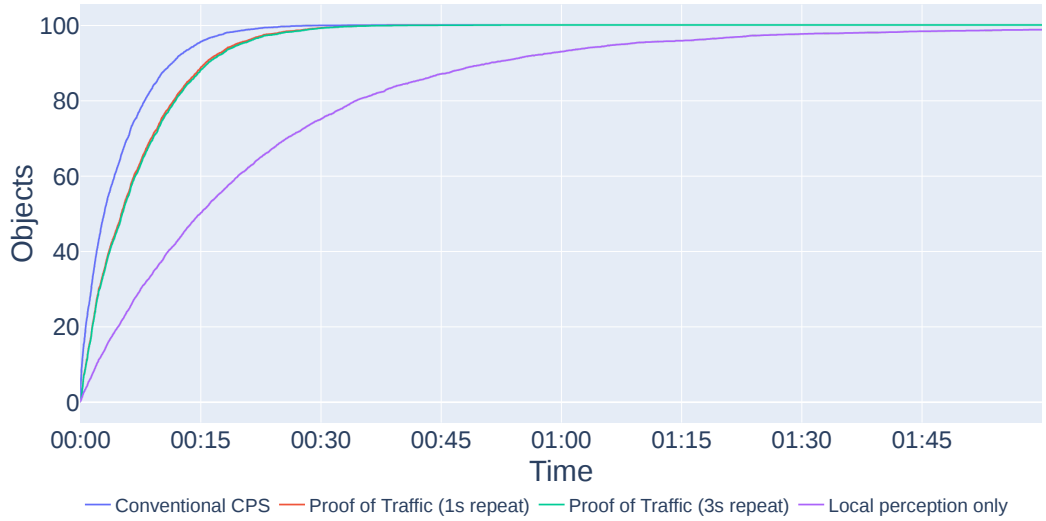


Figure 4.6: Accumulated available perceived objects (N_a) in Manhattan scenario

on this road, which has been highlighted in Fig. 4.7, and will be discussed later.

We have selected the same vehicle dimensions, communication and perception parameters, and then added additional parameters for extended vehicle behaviors, such as pseudonym changing. Flowsim is further optimized to ensure that the full-sized LuST scenario (i.e. 86400 ticks) could be finished in a wall time of one day.

Traffic density and the problem

Fig. 4.8 depicts the number of active on-road vehicles during the day. In the experiment, there are three rush hours in total: 10:00, 13:00, and 19:00. Thus, we selected three samples to analyze the different behaviors of the system under different scenarios: 7:00, i.e., before the morning rush hours; 12:00, i.e., at noon; 19:00, i.e., during the peak of evening rush hours.

Fig. 4.9 depicts the traffic density distribution in the three samples. In the heatmap of 7:00, the data points form a notable U-shaped path, representing the normal traffic on the E44. However, in the sub-figure of 12:00 and 19:00, we can see the data points on this road are very sparse, particularly in the southeast region, i.e., the bottom right area. This scarcity is caused by the complete congestion observed at the roundabout shown in Fig. 4.11; refer to the marked part in Fig. 4.7. In Fig. 4.11, the color of the lane represents their occupancy. Notably, the occupancy at E44 appears to be

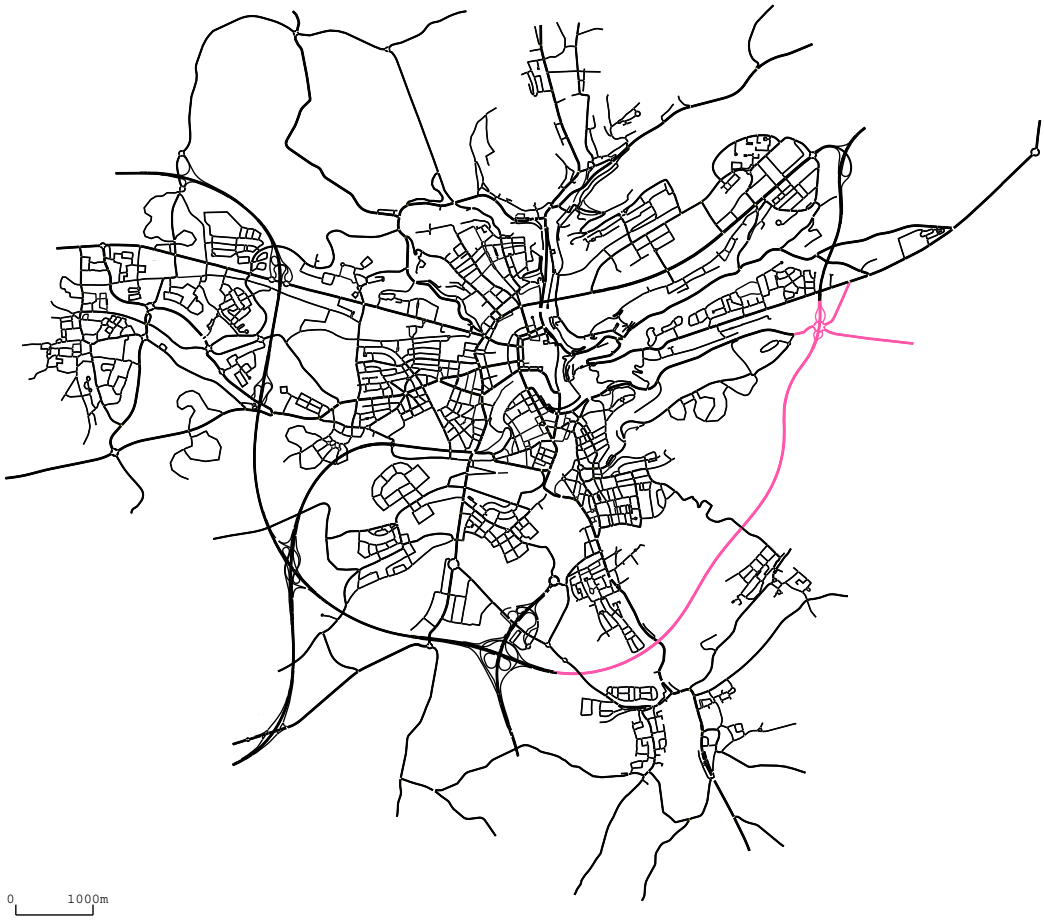


Figure 4.7: LuST scenario with the position of the E44 and the roundabout

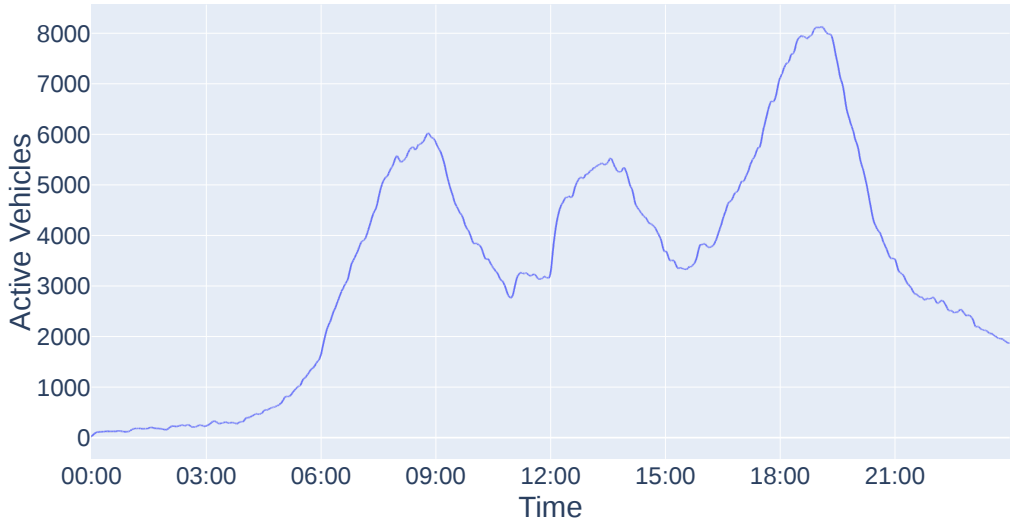


Figure 4.8: Number of active vehicles in the LuST scenario

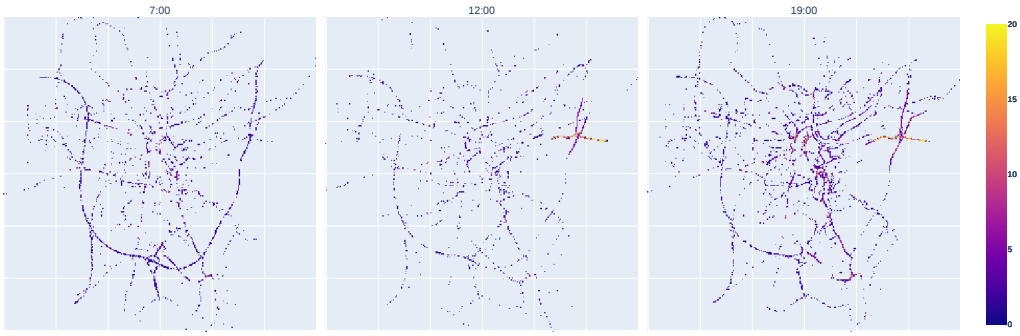


Figure 4.9: Number of vehicles in Luxemburg city at different times

very low, while the roundabout and the entrances/exits of E44 are completely blocked by vehicles. This severe congestion has even produced a large number of teleportation events [Doc], where vehicles approaching the roundabout from other roads were instantaneously teleported to different locations. The congestion prevents vehicles from entering E44, leading to missing data in the southeast corner of the road during those time intervals in Fig. 4.9 and 4.10.

We believe that this is caused by the well-known border traffic issue and the teleportation issue of SUMO scenarios; however, we have not modified the LuST scenario itself to ensure that it corresponds with that of other researchers. Nevertheless, we can still analyze the other part of E44 to observe behavior in similar highway traffic situations.

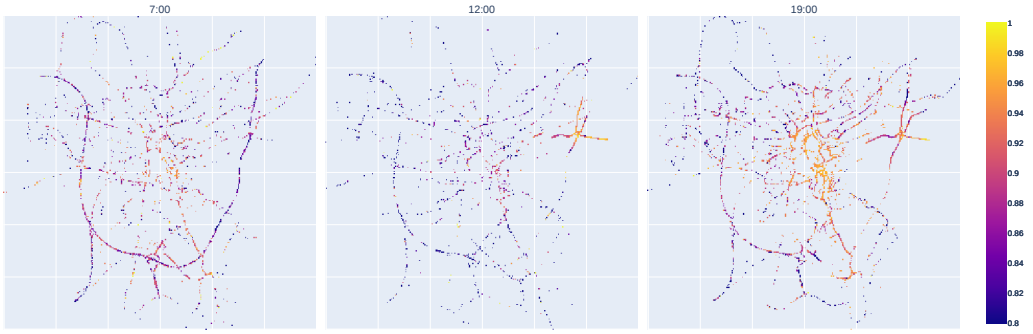


Figure 4.10: Verification ratio in Luxemburg city at different times

Verification ratio

The verification ratio is the main performance metric of the system. We measure the long-term verification ratio throughout the entire day, which concurs more with the real-world conditions, instead of calculating the verification ratio (named as success ratio in the last work) separately for each tick like in the last work.

Fig. 4.10 depicts the verification ratio in the same sampling time. We can observe that the verification ratio has a positive correlation to vehicle density. It starts at approximately 80 % at 7:00 and peaks at over 96 % at 19:00. Furthermore, the vehicles in the downtown area generally perform better than the ones on the highway. This is because the vehicles on the highway are typically platooning and did not encounter different vehicles for a relatively long period. Conversely, the diversity of roads in the downtown area and the randomness of the vehicle behaviors in that area make the platoons of vehicles very volatile, thus increasing the cooperative perception and the verification ratio. This phenomenon is particularly evident in the figure indicating the scene of 19:00.

Time-to-Verification

The timeliness of the cooperative perception data, i.e. the TTV, is the second thing to be considered. Fig. 4.12 depicts the TTV distribution of every hour, scaled to 100 %. We can observe that the system is bootstrapping, and that the traffic is sparse during the period from 0:00 to 7:00, resulting in longer verification delays. From the data after 9:00, we can observe that the trend becomes relatively flat, indicating that the system has reached dynamic equilibrium. Based on the trend, we can observe that approximately 35 % of the data can be verified in the same tick that it first appears, which

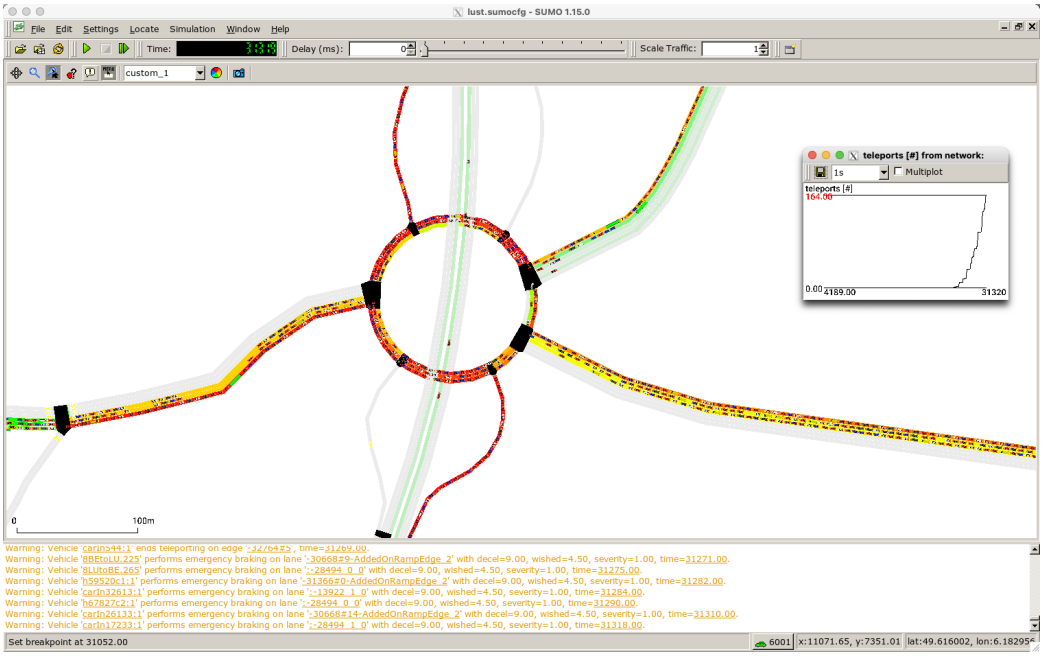


Figure 4.11: Vehicles and lane occupancy near the E44 roundabout at 08:42

produces a sub-second verification delay. Furthermore, over 65 % and 80 % of the observations could be verified within 1 s and 2 s, respectively. The result indicates that the proposed zk-PoT presents a very short latency from the point of receiving data to considering them trustworthy. Conversely, the conventional statistics and plausibility approaches require a significantly longer time to collect "evidence," to build the trust of either the sender or the received data.

Protocol overhead

Lastly, we must consider the protocol overhead as well. Fig. 4.13 shows that the system is not fully bootstrapped in the simulation time before 9:00, similar to the previous pattern. Beyond this point, we can consider it as the normal working condition of our system. Therefore, we primarily focus on analyzing the differences in this phase.

In the conventional CPS with no PoT enabled, the average bandwidth consumption of a single vehicle stabilizes at approximately 6 kbps. In the PoT-enabled system with proofs repeated every second, the system presents a 40 % increase in bandwidth when compared with the baseline system, reaching 9 kbps. However, if we change the PoT to repeat proofs every 3 s, the bandwidth increment reduces to about 25 %, resulting in a bandwidth

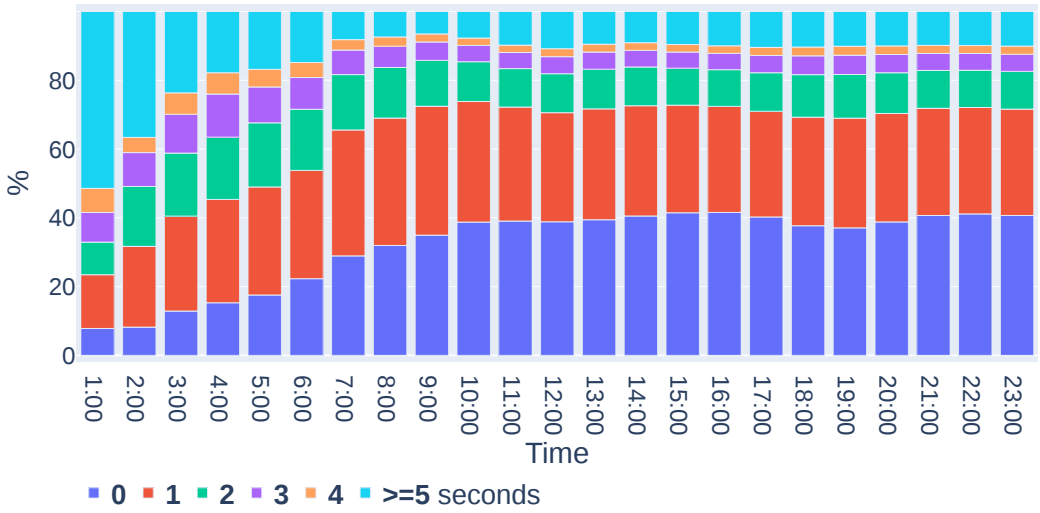


Figure 4.12: Time-to-Verification (TTV) distribution

of 7.5 kbps, which is considered stable and feasible. Furthermore, the bandwidth did not fluctuate according to the total number of vehicles, which is attributed to the dynamic equilibrium and saturation effects.

4.4 Threat Analysis

The zk-PoT provides vehicles with the capability to swiftly and deterministically validate data received from the CPS. This eliminates the possibility of naïve attacks and data tampering, such as fabricating non-existent vehicles and duplicating existing vehicles. Furthermore, various complex attacks can be prevented or mitigated. Although this list is not exhaustive, it covers some of the most common attacks that can be encountered.

4.4.1 Brute-Force Attack and Dictionary Attack

Malicious entities may attempt to perform brute-force attacks or dictionary attacks on the number plate used in the proof.

In the case of a brute-force attack, the attacker can try to match the public keys used in other vehicles' proofs by only "hearing" the candidate observed vehicles. As the range of "hearing" is considerably larger than that of "seeing," attackers can still pretend that they "saw" the target vehicles. Furthermore, this attack can be done in an offline fashion and does not require any exchange of information with other vehicles. We can eliminate these types of attacks by employing key derivation functions (KDFs), such

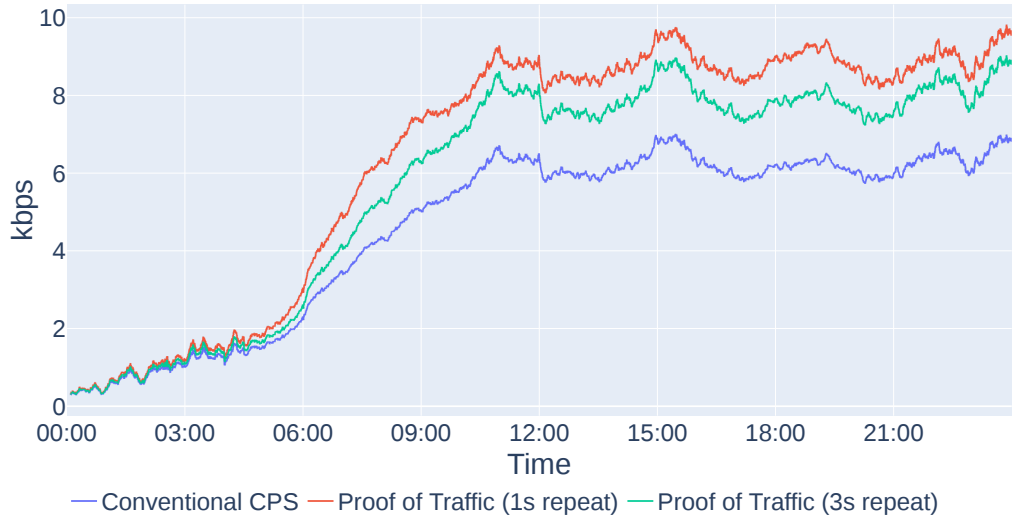


Figure 4.13: Bandwidth consumption per vehicle

as PBKDF2 [KR17] or scrypt [PJ16], to increase the length of time required for a successful brute-force attack, making it longer than the lifespan of a pseudonym.

In the case of a dictionary attack, the attacker can gather previously known number plates and use them as a reference. This type of attack presents a higher success rate than that of a brute-force attack due to the locality of the vehicles. Mechanics like ephemeral salt can be employed to prevent this attack.

4.4.2 Location Privacy Against Tracking

The use of pseudonyms is crucial in ensuring location privacy in V2X communications. With the combination of pseudonyms and KDFs involved, brute-force attacks cannot be performed to create a timely proof. However, an attacker can still collect data on the road and recover a vehicle's trajectory.

Fortunately, after the target changes its pseudonymous ID, remote attackers can no longer track it, even if they have information about its old ID and number plate. As such, the proposed mechanism provides at least the same level of privacy as that of traditional pseudonyms.

4.4.3 Spam Attack

Verifiers are vulnerable to spam attacks as partial proofs cannot be falsified, Attackers can easily create seemingly valid proofs randomly, which can

deplete the victim verifiers' computing power and memory, thus potentially causing a denial-of-service situation.

This type of attack can be mitigated by setting a limit on the number of unmatched proofs from each prover. If a vehicle exceeds this limit, the verifier must disregard any subsequent proofs sent by the vehicle until some of them are matched by other vehicles.

4.4.4 Signal Jamming Attack

In the proposed scheme, a malicious vehicle cannot simply replay a message. However, the attacker can still jam the signal from the original prover and then replay its proof as their own.

This kind of attack can be mitigated by including the pseudonym of the prover in the proof. This is because the pseudonym of a vehicle cannot be easily forged, making it a secure method of identifying the original prover.

4.4.5 Collusion Attack and Sybil Attack

Collusion and Sybil attacks are both types of attacks that are specific to the applications that require voting or cross-verification, including the proposed method. A collusion attack is a type of attack in which multiple malicious entities work together to deceive a system. In a Sybil attack [Dou02], the attacker creates multiple pseudonymous identities to impersonate multiple users. These two types of attacks can be combined to create a set of fake vehicles that can deceive victim verifiers.

While these attacks are typically challenging, we can implement certain strategies to mitigate them. We can mitigate the Sybil attack by implementing mechanisms that prevent the simultaneous use of different pseudonyms belonging to the same vehicle. The design of pseudonyms is not directly related to the proof system and will be addressed in the future. For the non-Sybil collusion attack, the number of vehicles controlled by the attacker is limited as they require a fleet to perform the attack. In this case, we can limit the revenue from the data. The revenue could comprise a trust value or credits, and it varies depending on the system built based on the proof system. For example, we can set the trust value for a successful match between two specific provers to be the reciprocal of the number of accepted matches over the past 15 minutes. This countermeasure can cause a sharp decrease in trust, making further collusion unprofitable.

4.5 Summary

This study proposed a novel deterministic cross-verification scheme called zk-PoT. By enabling vehicles to generate zero-knowledge blind proofs to their independent observation, the remote parties can be convinced by cross-verifying these proofs, without leveraging the ground truth or trust evaluation. Subjecting to the zero-knowledge property, the zk-PoT will not reveal any information about any particular vehicle, thereby preserving the location privacy of the vehicles.

The quantitative simulation analyses in the Luxemburg scenario verify that zk-PoT is in good performance. In the experiments, about 80 % to 96 % of observed vehicles can be cross-verified by zk-PoT. Over 80 % of cross-verification happened within a sub-2 s delay and over 90 % happened within 5 s. The bandwidth consumption overhead is approximately 25 % compared to the original 1 Hz CPS standard.

The threat analysis against various kinds of attackers shows that zk-PoT could survive most of the common threats, such as brute-force attacks, dictionary attacks, spam attacks, and signal jamming attacks. It can maintain location privacy to the same level as the original pseudonym system. The collusion attack is challenging, but it can still be mitigated by extra rules.

Zk-PoT can be either used as a standalone method or integrated with existing cooperative perception standards, as shown in the aforementioned example, where zk-PoT is implemented for the CPS in the ETSI/ITS standards. Furthermore, zk-PoT is particularly helpful for bootstrapping the trust establishment process in several existing trust management models as they can provide strong evidence that can be cross-verified by other vehicles [HLZ21].

Owing to the limited scope of this study, not all aspects have been examined comprehensively. The following issues should be considered in the future. Firstly, the vehicles are not incentivized to share data and tend to be selfish. This will reduce the number of proofs thus harming the cross-verification ratio. Based on the "privacy-trust dilemma," an economic model could be proposed. Secondly, Sybil attacks are not solved even when collusion mitigation techniques are introduced. This is a potential research direction to enhance zk-PoT. Lastly, the non-connected vehicles are not considered in this study because they lack identity with enough entropy. Some other mechanics should be considered to make zk-PoT better adapted to mixed traffic environments.

Chapter 5

Zero-Knowledge Proof of Distinct Identity

Pseudonyms are widely used in C-ITS to protect the location privacy of vehicles. However, the unlinkability nature of pseudonyms also enables Sybil attacks, where a malicious vehicle can pretend to be multiple vehicles at the same time. In this paper, we propose a novel protocol called zero-knowledge Proof of Distinct Identity (zk-PoDI,) which allows a vehicle to prove that it is not the owner of another pseudonym in the local area, without revealing its actual identity. zk-PoDI is based on the Diophantine equation and zk-SNARK, and it does not rely on any specific pseudonym design or infrastructure assistance. We show that zk-PoDI satisfies all the requirements for a practical Sybil-resistance pseudonym system, and it has low latency, moderate computation overhead, and negligible communication cost. We also discuss the future work of implementing and evaluating zk-PoDI in a realistic environment.

5.1 Problem Statement, Assumptions and Requirements

In the last section, we reviewed the current progress of the pseudonym systems and Sybil attack countermeasures in C-ITS, with their reasons against deployment respectively. In order to fill the gap in pseudonyms against the Sybil attack, in this section, we will formalize our particular problem regarding the Sybil attack, make a few reasonable assumptions about the underlying pseudonym system, and finally detail our requirements for the proposed work.

5.1.1 Problem statement

To enhance the efficiency of road transportation, we aspire for vehicles within a local area to exchange information and report their respective situations to one another. Each vehicle should utilize multiple pseudonyms with no obvious connections between them to safeguard the owner's privacy from potential leaks.

Let us take a vehicle, Alice. It will receive numerous messages sent by different pseudonyms. Alice can adjust its driving strategy based on these messages. However, since Alice cannot establish any connections between the pseudonyms, it must be cautious of the possibility of a dishonest vehicle using various pseudonyms to send multiple messages, attempting to gain an advantage.

Alice needs a method to identify the relationship between such identities. The most straightforward approach is to send a request to the LEA to confirm that the two received pseudonyms indeed originated from distinct vehicles. However, maintaining constant contact with LEA is impractical, and such inquiries may compromise privacy. For these reasons, we must avoid making the identification process dependent on LEA involvement.

In accordance with the aforementioned requirements, we propose that vehicles generate proof and include it with their messages. The purpose of this proof is to enable Alice, upon receiving two messages from different pseudonyms, to easily verify that these messages were genuinely sent by distinct vehicles rather than two pseudonyms of a single dishonest vehicle. Conversely, if a dishonest vehicle sends two messages with different pseudonyms, the proofs accompanying these messages would be rejected by Alice, leading to the rejection of the messages.

In our paper, we employ the Diophantine equation to formulate the cryptographic protocol. As mentioned earlier, solving the Diophantine equation is a computationally challenging task, prompting us to use it as the pseudonym for a vehicle. The LEA generates a set of Diophantine equations with identical solutions and assigns them all to a vehicle as its pseudonyms.

When Charles receives a packet from Alice, it notes that Alice's pseudonym $\mathcal{F}$. Meanwhile, another vehicle, referred to as Bob and denoted by the pseudonym $\mathcal{G}$, is also transmitting a message. Alice receives $\mathcal{G}$ and generates a proof asserting their distinctiveness. Specifically, in the proof, Alice claims that $\mathcal{F}$ is not another pseudonym of Bob, thereby demonstrating that they are distinct vehicles. Alice also needs to make sure that the proof would not leak their pseudonym. The pseudonym $\mathcal{F}$ is signed by the LEA, and Charles trusts that Alice cannot independently generate the pseudonym $\mathcal{F}$.

Considering another pseudonym $\mathcal{F}'$ associated with Alice, Charles cannot

ascertain that it is an additional pseudonym of Alice, as it cannot deduce the solutions of both pseudonyms and identify that they share the same solution.

The security of the aforementioned process is ensured by the implementation of zero-knowledge proofs.

5.1.2 Assumptions

At the time of writing, the pseudonym is not yet standardized by either ISO or ETSI. To maintain maximum compatibility with the future, we only make the following two minimum assumptions about the underlying pseudonym system.

Firstly, the pseudonym design should meet the basic common requirements. Specifically, it should be cryptographically strong enough and have a decent unlinkability.

Secondly, the pseudonym should be centralized issued, i.e. by LEA. Consequently, the pseudonym will be signed by LEA, with its own public key infrastructure, commonly known as the pseudonym CA.

Lastly, the secret key of pseudonyms should not be extracted or left to the vehicle issued. This is important because if they can be extracted, then an attacker could gather *permanent* identities from other vehicles with physical access, which is not considered a Sybil attack and thus out of this paper's scope. This should be ensured by hardware-based security components, such as HSM and anti-tamper OBU.

5.1.3 Requirements

In order to accommodate our work into the existing C-ITS standard framework, some requirements should be met. A recent survey about detecting and preventing Sybil attacks for C-ITS [HIZo22] points out 4 major "do not" requirements for a Sybil-resistant system:

- Do not modify the existing C-ITS standards.
- Do not break current C-ITS security architecture.
- Do not rely on vehicles' history.
- Do not break the unlinkability of the original pseudonym.

Also, it points out several features which are considered adequate for adoption in C-ITS standards:

- Do not require additional hardware.

- Considers C-ITS standards.
- Feasible on current C-ITS systems.
- Do not rely on the digital map.
- Do not allow tracking by the public.
- Scalable to large scenarios.

All the above requirements are crucial for the work to be integrated into the standards. Moreover, we also summarized some additional requirements which are considered equally important:

Non-interactive: do not require interaction with infrastructure (LEA or RSUs), nor the other vehicles. The former is important because we cannot expect real-time global connectivity in every situation in C-ITS. The latter is also considered good because interacting with other vehicles will increase the traffic and potentially harm the overall decision time.

Succinct: the work should maintain relatively small communication overheads and low processing footprints for the vehicles.

5.2 Mathematical Primitives

Before diving into the actual solution, in this section, we will introduce some mathematical tools to help solve the problem. The first one is "the integer solution of a Diophantine equation", which is considered as the "computational hard problem" in the proposed cryptography protocol. The second one is zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARK), a set of constructions that can be used to convert the first problem into a zero-knowledge proof.

5.2.1 Diophantine Equation and its Integer Solution

A Diophantine equation is a type of equation involving two or more unknowns and integer coefficients. The general form of a Diophantine equation is given as:

$$\sum_{i=1}^{k} a_i x_1^{b_{i1}} \dots x_n^{b_{in}} = 0. \tag{5.1}$$

Where $a_1, \dots, a_n$ are integer, $b_{11}, \dots, b_{nn}$ are non-negative integer.

For a given Diophantine equation, let $d = \max \sum_j b_{ij}$ for all $0 < i \leq k$. We refer to such an equation as a d-degree Diophantine equation. If $\sum_j b_{1j} = \cdots = \sum_k b_{kj}$ holds, we term it a homogeneous Diophantine equation.

Solving Diophantine equations is trivial when considering solutions in a general sense. However, finding integer solutions to these equations is an exceptionally challenging problem. In the context of this paper, whenever we discuss solutions to Diophantine equations, we are exclusively referring to integer solutions.

The difficulty of solving Diophantine equations increases sharply with the increase in degree. For fourth-degree or higher Diophantine equations, determining whether a solution exists becomes Hilbert's tenth problem, which was solved in 1970 by Matiyasevich. The answer is negative: It is impossible to devise a universal method for such determination.

Solving a first-degree Diophantine equation is relatively straightforward, following the generalized Bézout's lemma. The equation in the form of $\sum_{i=1}^{k} a_i x_i$ has a solution if and only if $\gcd(a_1, \ldots, a_n)$ divides the constant term c.

The complexity increases with second-degree Diophantine equations, where solutions may yield large numbers even for small coefficients. Moving to third-degree equations, the difficulty is further increased, but we can still solve homogeneous 3-order Diophantine equations using elliptic curves

However, for fourth-degree and higher equations, Hilbert posed a fundamental question as the tenth of Hilbert's problems: Is there a general algorithm to decide whether a Diophantine equation has integer solutions? Unfortunately, the answer is negative, as shown by Matiyasevich's theorem. Due to space constraints, delving into the specifics and proof of the theorem is omitted.

Solving a 1-degree Diophantine is relatively straightforward. According to generalized Bézout's lemma, a 1-degree Diophantine equation in the form of $\sum_{i=1}^{k} a_i x_i$ can be solved if and only if $\gcd(a_1, \ldots, a_n)$ divides the constant term c.

The 2-degree Diophantine equation is significantly more challenging than the 1-degree case. Solutions can yield large numbers, even when the coefficient k is small. For example: The Diophantine equation $y^2 - y - x^2 + x - 2xy = 0$ have a solution $x = 1583407981$ and $y = 3822685023$.

When extending the problem to 3-degree Diophantine equations, the difficulty increases. For a 3-degree homogeneous Diophantine equation, it is feasible to find the solution solutions by transforming the equation into an elliptic curve form.

However, as we move to the 4 degree and higher, the landscape changes. Hilbert posed a fundamental question as the tenth of Hilbert's problems: Is

there a general algorithm to decide whether a Diophantine equation possesses integer solutions? Regrettably, the answer is negative. It is impossible to devise a universal method for such determination. This conclusion directly follows from Matiyasevich's theorem. Given space constraints, we would not delve into the specifics and proof of the theorem here.

Although finding integer solutions for higher-degree Diophantine equations is extremely hard. Nevertheless, constructing a Diophantine equation with integer solutions is relatively straightforward. Any set of integers can be chosen as the unknowns, and a Diophantine equation with solutions can be generated. Furthermore, verifying whether a set of numbers constitutes a solution to a Diophantine equation is also a trivial task. In other words, the problem of finding solutions to Diophantine equations belongs to the NP-problem. This insight has inspired the use of Diophantine equations in cryptographic designs.

5.2.2 zk-SNARKs

In this section, we will introduce a type of zero-knowledge proof known as Zero-Knowledge Succinct Non-Interactive Argument of Knowledge, abbreviated as zk-SNARKs. By using zk-SNARK, the prover can show the verifier that he has a solution to an equation without having to reveal what the solution is.

In chapter 2, we presented an overview of zero-knowledge proofs. In 1991, building upon the foundational work of [GMR89], Blum, Feldman, and Micali introduced non-interactive zero-knowledge (NIZK) proofs in their seminal paper [BDSMP91]. In NIZK proofs, the prover only needs to transmit a single message to the verifier to fulfill the necessary proof requirements.

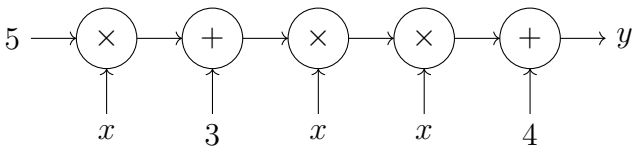
The following outlines the construction process of zk-SNARKs. Firstly, we articulate the problem that we aim to prove. Consider a polynomial:

$$f(x) = a_1 x^p + a_2 x^{p-1} \dots a_{p-1} x + a_p. \tag{5.2}$$

For a specific set of coefficients $a_1 \dots a_p$ and a constant N, the prover seeks to convince the verifier that they indeed possess a solution k such that $f(k) = N$ without disclosing the exact value of k.

Arithmetic Circuit: An arithmetic circuit can be derived from a polynomial. It is represented as a directed acyclic graph with addition and multiplication gates as vertices and wires as edges. Each vertex has two input wires and one output wire, with the entire circuit having only one output wire.

For instance, a graph can be generated from the polynomial $5x^3 + 3x^2 + 4$ as follows:



R1CS: Rank-1 Constraint System, or R1CS, provides an alternative representation of the polynomial. Specifically, R1CS rewrites an arithmetic circuit that expresses polynomials into a set of equations. By using R1CS, we can algebraic the arithmetic circuit for further encryption operations.

For example, the R1CS of polynomial $5x^3 + 3x^2 + 4$ will be:

$$\begin{aligned} y &= S_4 + 4 && (5.3)\\ S_4 &= S_3 \cdot x && (5.4)\\ S_3 &= S_2 \cdot x && (5.5)\\ S_2 &= S_1 + 3 && (5.6)\\ S_1 &= x * 5 && (5.7) \end{aligned}$$

Quadratic Arithmetic Program (QAP): QAP is a method for transforming R1CS into a set of vector operations. Initially, we can represent R1CS as a vector $\mathbf{s}$ which constructed by the R1CS.

$$\mathbf{s} = (1, x, y, S_1, \ldots, S_n)^{\mathrm{T}}. \tag{5.8}$$

Here, $S_1, \ldots, S_n$, and y are components of the R1CS. We refer to $\mathbf{s}$ as a witness vector.

Assuming there are n wires and m gates in the arithmetic circuit, we construct three $n \times m$ matrices A, B, C such that $A\mathbf{s} \odot B\mathbf{s} = C\mathbf{s}$. Here, $\odot$ denotes the Hadamard product.

Then, we are able to use the Common Reference String (CRS), homomorphic encoding and linear pairing to construct the protocol of zk-SNARK.

$$A = \begin{pmatrix} \mathbf{a}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{a}_n^{\mathrm{T}} \end{pmatrix}, B = \begin{pmatrix} \mathbf{b}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{b}_n^{\mathrm{T}} \end{pmatrix}, C = \begin{pmatrix} \mathbf{c}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{c}_n^{\mathrm{T}} \end{pmatrix}, \tag{5.9}$$

The follow-up procedure is described in [Gro16].

In short, zk-SNARK is a cryptographic protocol that allows a prover to convince a verifier that a given statement is true, without revealing any additional information, in a short and efficient manner. A zk-SNARK protocol

consists of three polynomial-time algorithms: a key generator $\mathcal{G}$, a prover $\mathcal{P}$, and a verifier $\mathcal{V}$, defined as follows:

The key generator $\mathcal{G}$ takes as input a security parameter λ and a program C, and outputs a pair of public keys: a proving key pk and a verification key vk. These keys are public parameters that can be reused for multiple proofs for the same program C.

The prover $\mathcal{P}$ takes as input the proving key pk, a public input x and a private witness w. The algorithm outputs a proof $\pi = \mathcal{P}(pk, x, w)$ that attests that the prover knows a witness w such that $C(x, w) = \text{true}$.

The verifier $\mathcal{V}$ takes as input the verification key vk, the public input x and the proof π. The algorithm outputs a binary value $\mathcal{V}(vk, x, \pi) \in \{0, 1\}$ that indicates whether the proof is valid or not. The verifier accepts the proof if and only if $\mathcal{V}(vk, x, \pi) = 1$.

5.3 Proposed Method

In subsection 5.2.1, we have introduced the mathematic open problem, the Diophantine equation. Based on the Diophantine equation, we are able to construct a cryptographic system to meet our demands.

Recall that for a high-degree Diophantine equation, finding the solution of it is extremely difficult. However, if we already found the solution, verifying whether the solution is true is easy.

As an example, Consider a 8-degree Diophantine equation with 4 unknown numbers:

$$\mathcal{D}(\vec{a}, \vec{x}) = a_1 x_1^8 + a_2 x_2^8 + a_3 x_3^8 + a_4 x_4^8. \tag{5.10}$$

Where $\vec{a} = (a_1, a_2, a_3, a_4)$ and $\vec{x} = (x_1, x_2, x_3, x_4)$. For simplicity of implementation and discussion, we decided to use a Diophantine equation with no mixed terms, and the degrees of all unknowns are the same power of 2.

By choosing different coefficients $\vec{a}$ we are able to generate different Diophantine equations. For example, we choose an arbitrary set of coefficient $\vec{m} = (m_1, m_2, m_3, m_4)$, then we obtain a Diophantine equation $f(\vec{x}) = \mathcal{D}(\vec{m}, \vec{x})$.

For an arbitrary set of integers $\vec{s} = (s_1, s_2, s_3, s_4)$, let $y = f(\vec{s})$, we can generate a new Diophantine equation

$$\mathcal{F}(\vec{s}) = m_1 s_1^8 + m_2 s_2^8 + m_3 s_3^8 + m_4 s_4^8 - y = 0. \tag{5.11}$$

By doing so, we can obtain an equation with $\vec{s}$ as a solution arbitrarily. However, it is difficult to solve it to obtain the solution, thus, we can believe that the one who can provide the solution must be the generator of the function.

In the same way, if we choose a different set of coefficients $\vec{n} = n_1, n_2, n_3, n_4$, there will be a different Diophantine equation $g(\vec{x}) = \mathcal{D}(\vec{n}, \vec{x})$. Using the same $\vec{s}$, we obtain $z = g(\vec{s})$ and another Diophantine equation

$$\mathcal{G}(\vec{s}) = n_1 s_1^8 + n_2 s_2^8 + n_3 s_3^8 + n_4 s_4^8 - z = 0. \tag{5.12}$$

After the above process, we obtain two Diophantine equations $\mathcal{F}$ and $\mathcal{G}$, which seem unrelated, except that they are both difficult to solve, and only the generator of the equation knows that $\mathcal{F}$ and $\mathcal{G}$ share the same solution $\vec{s}$.

Consider another Diophantine equation $\mathcal{H}(\vec{x})$ which belongs to another vehicle, it is easy to verify that $\mathcal{H}(\vec{s}) \neq 0$ since that $\mathcal{H}$ is generated by a set of integers which different with $\vec{s}$.

By using the above method, we can generate any number of Diophantine equations with known solutions. The solution s_1, s_2, s_3, s_4 is the fixed and unique identity of the creator or prover. For the verifier, we need to construct a method such that they can believe that the prover has the solution of a Diophantine equation, and the solution is not another equation's solution. In other words, they need to verify that the prover has a solution $\vec{s} = (s_1, s_2, s_3, s_4)$ of the following set of equations:

$$\begin{cases} \mathcal{F}(\vec{s}) = 0 \\ \mathcal{G}(\vec{s}) \neq 0 \end{cases} \tag{5.13}$$

We refer to it as the **distinct identity criterion (DIC)**.

Based on the difficulty of solving such an equation, if the prover indeed has the solution, the verifier should believe that the prover is the creator of the equation.

Notice that, since the pseudonyms have enough entropy, we can let every party generate the $\vec{a}$ via a determined algorithm (e.g. cryptographic hash) on the fly, to get rid of transferring them along with the pseudonyms. The only variable that needs to be transferred (and sealed) is the constant term y.

Next, to enable a prover to show that it possesses the solution s_1, s_2, s_3, s_4 but without directly revealing the solution themselves, the equations in "plaintext" should be converted to an arithmetic circuit, which can in turn be converted to prove function (PF) and verify function (VF) of zk-SNARK.

The overview of the corresponding arithmetic circuit is depicted in Figure 5.1.

Function $Pow(s, e)$ is to calculate the s^e with integer s and e using the double-and-add algorithm, which is detailed in Algorithm 2. In the situation of the degree of Diophantine equations being a power of 2, the total number of multiplication gates will be $log_2 e$.

Function $Dot(A, B)$ is the trivial dot product circuit, which calculates the pairwise product of each item in two vectors and sums all the products together. The number of multiplication gates is $|A|$.

Function $Equal(a, b)$ is directly translated to a multiplication gate in R1CS, namely $1 \cdot a = b$.

Function $NotEqual(a, b)$ requires a slightly complicated construction and can be translated to a multiplication gate with the form of $v \cdot (a-b) = 1$. This is possible because: if $a \neq b$, there must be a free variable v which satisfies the constraint; if $a = b$, then $a - b = 0$, and nothing can make $v \cdot 0 = 1$.

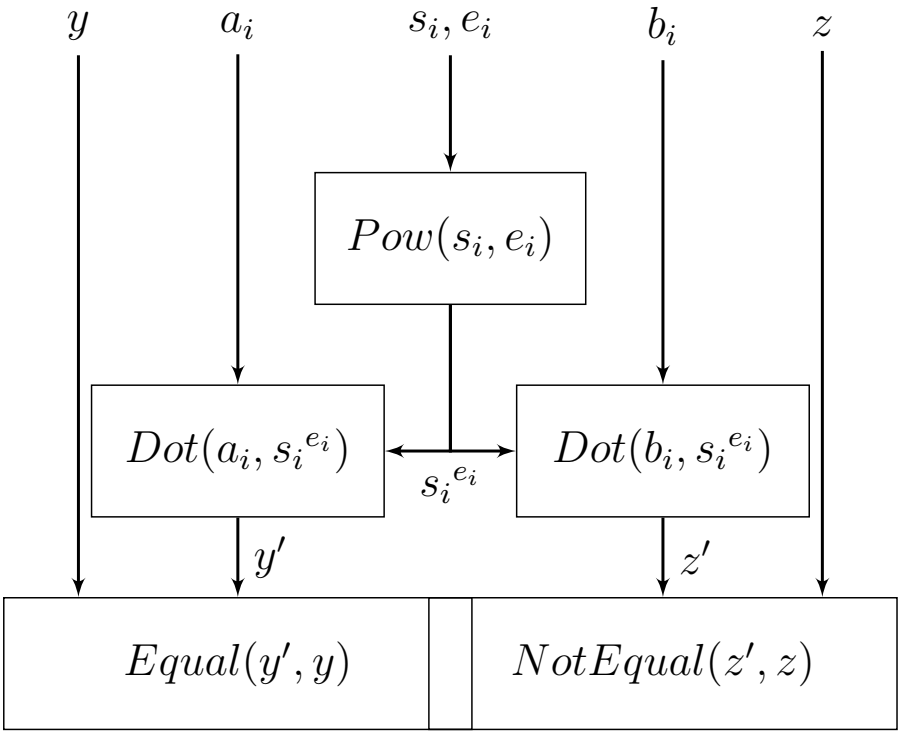


Figure 5.1: Arithmetic circuit overview

Utilizing the DIC, protected by zk-SNARK, we finally propose the actual cryptographic protocol, called **zero-knowledge Proof of Distinct Identity (zk-PoDI)**, which is outlined as follows:

Algorithm 2 Fast Power Circuit Pow

```
1: procedure Pow(s, e)
2:     Unpack e into a vector evec
3:     out = 1
4:     while j < len(evec) do
5:         if evec[j] == 1 then
6:             out = out * s
7:         s = s * s
8:         j = j + 1
       return out
```

Setup, done by LEA:

1. Global setup:
   Pick a complex enough form of Diophantine equations $\mathcal{D}$. Express the corresponding DIC in the language of R1CS and calculate the corresponding arithmetic circuit C. Generate the proof function PF and verify function VF based on C with the corresponding keys (PK, VK), based on zk-SNARK. Distribute (PK, VK) to every vehicle.

2. Generate the orthonym for each vehicle:
   Pick a set of cryptographic random numbers $\vec{s} = (s_1, \ldots, s_n)$ as the prover's orthonym. Distribute each orthonym to the vehicles.

3. Generate each pseudonym of each vehicle:

   (a) Generate a sufficient list of pseudonyms $P_1, \ldots$ for the vehicle, using the underlying pseudonym generation algorithm.

   (b) Generate coefficients $\vec{m_k} = (m_{k1}, \ldots, m_{kn})$ based on to the pseudonym P_k of the vehicles. The particular generation algorithm is not limited as long as it is strong enough, e.g. using cryptographic hash functions.

   (c) For a pseudonym P_k, plug in the $\vec{s}$ and $\vec{a}$ into quiz generation method to generate the quiz equation $\mathcal{F}(x) = 0$. It is determined by P_k and $y = f(\vec{s})$ so everyone can reproduce $\mathcal{F}$ according to them.

   (d) Sign each pair of P_k and y_k with the pseudonym CA and distribute them to the vehicle.

Prove:

1. Given the current pseudonym P_e, y_e of the egovehicle, and the signature-verified pseudonym P_k, y_k of another vehicle.
2. Generate coefficients $\vec{a}_e$ according to the pseudonym P_e; generate $\mathcal{F}$ using $\vec{a}_e$ and y_e.
3. Do the same with $\vec{a}_k$ and y_k to obtain $\mathcal{G}$.
4. Calculate $\pi = PF(pk; \mathcal{F}, \mathcal{G}; \vec{s})$.
5. Publish π, P_e, y_e, P_k, y_k.

Verify:

1. Given π, P_e, y_e, P_k, y_k.
2. Verify the LEA signature of P_e, y_e and P_k, y_k.
3. Generate the quiz equations $\mathcal{F}$ and $\mathcal{G}$ according to the pseudonym P_e, P_k.
4. Accept if $VF(vk; \mathcal{F}, \mathcal{G}; \pi) = true$, otherwise reject.

Figure 5.2 depicts an overview of how the protocol works and what is transmitted from LEA to the vehicles, from prover to verifiers. The pk and vk are omitted in the figure. LEA creates the solution for each prover as its orthonym $\vec{s}$; then it creates one quiz equation $\mathcal{F}_1, \mathcal{F}_2, \cdots$ for each of its pseudonym, and distributes all of them to the vehicle with its signature (endorsement.) For a vehicle generating a proof, it plugs in its own $\vec{s}$, the quiz function of its current pseudonym, $\mathcal{F}_i$, and the function of the other vehicle, $\mathcal{G}_j$. The output is a value π, representing its proof of distinctiveness. Finally, for a verifier, it can be convinced that $\mathcal{F}_i$ and $\mathcal{G}_j$ are distinct vehicles by using only $\mathcal{F}_i$, $\mathcal{G}_j$ and π.

The zk-PoDI is considered a valid zero-knowledge proof system because it satisfies all three required properties:

Completeness:

If a dishonest prover conducts a Sybil attack, i.e. they use two different pseudonyms and pretend to be two different vehicles. In this case, they cannot construct a solution S such that the DIC is held, because the two equations are both true (the real solution) or both false (the non-solution).

Soundness:

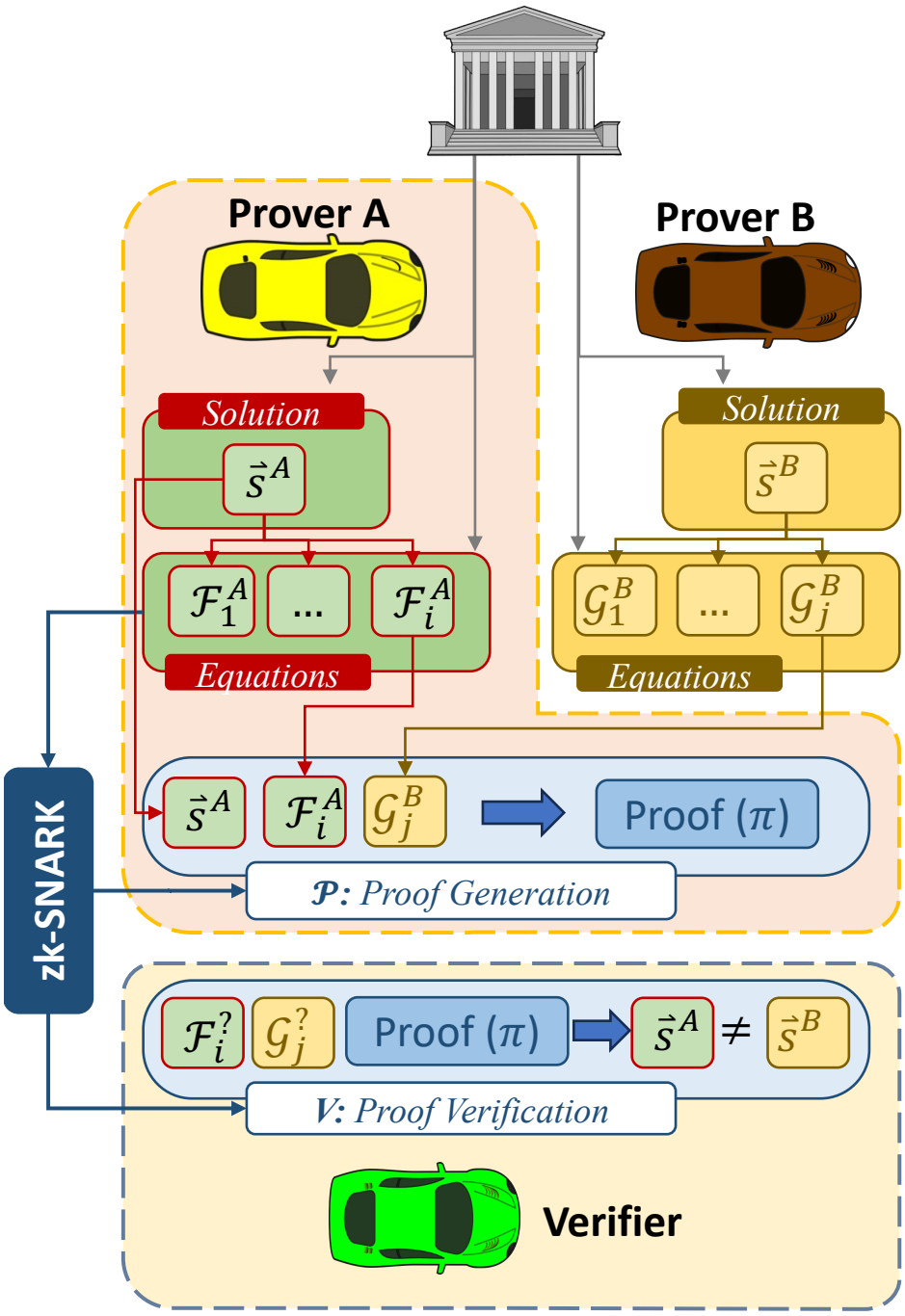


Figure 5.2: Protocol overview

Solving the Diophantine equation is an extremely difficult problem. With today's computational techniques, it is still impossible to solve a high-degree

Diophantine equation. With the increasing of the amount of unknown numbers, the probability of a random collision solution drops to close to zero.

The constant term y is signed by the LEA together with the pseudonym P. Although the coefficients $\vec{m}$ are generated on the fly, however, $\vec{m}$ is generated by the P which is still signed by the LEA. Therefore, a dishonest vehicle cannot construct a separate equation since the whole equation is sealed by LEA.

Finally, the existence of such $\vec{s}$, which indicates the prover indeed possesses the orthonym, is guaranteed by zk-SNARK.

Zero-Knowledge:

The protocol is zero-knowledge because the orthonym $\vec{s}$ is the secret input of zk-SNARK and thus will not be revealed. Instead, the proof π carries all the information needed to convince a verifier.

5.4 Discussion

5.4.1 Performance Analysis

Our protocol uses the Groth16 [Gro16] implementation of zk-SNARKs because of its minimum verifier and communication complexity. Groth16's major downside is it requires a pre-circuit setup procedure, which might be a problem in other applications. However, this is not a problem for our zk-PoDI, because it only requires one global circuit for DIC during the deployment of the whole pseudonym system.

Based on the previously mentioned assumptions, we can fix the form of DIC as Equation 5.14 and analyze the complexity of zk-PoDI with different numbers of equations (Nx) and numbers of power (Np). Here we limit the Np to be a power of 2, to achieve maximum efficiency in the double-and-add algorithm. By adjusting the super-parameters Nx and Np, we can evaluate how the strength of zk-PoDI against different attacks changes.

$$\begin{cases} \mathcal{F}(\vec{S}) = a_1 s_1^{Np} + a_2 s_2^{Np} + \cdots + a_{Nx} s Nx^{Nx} - y = 0 \\ \mathcal{G}(\vec{S}) = b_1 s_1^{Np} + b_2 s_2^{Np} + \cdots + b_{Nx} s Nx^{Nx} - z \neq 0 \end{cases} \tag{5.14}$$

In our implementation, we use the Alt-bn128 curve, which is a commonly used curve in e.g. the Ethereum blockchain. The finite fields $\mathbb{F}_q$ and $\mathbb{F}_{q^2}$ in Alt-bn128 is 254 and 508 bits long; the element of $\mathbb{G}_1$ is $\mathbb{F}_q$ plus a sign bit; the element of $\mathbb{G}_2$ is $\mathbb{F}_{q^2}$ plus a sign bit; and the proof consists of two elements in $\mathbb{G}_1$ and one in $\mathbb{G}_2$. Thus, the total size of the proof π is given by:

$$|\pi| = 2|\mathbb{G}_1| + |\mathbb{G}_2| = 2(254 + 1) + 508 + 1 = 1019\text{bits}$$

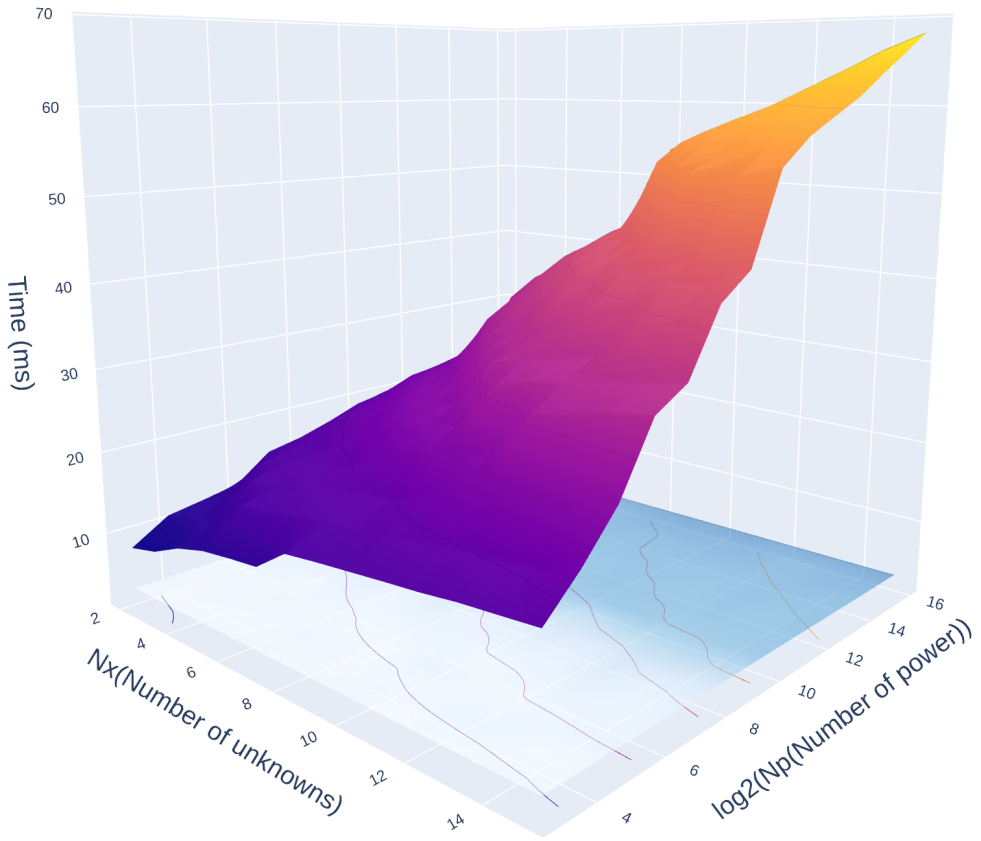


Figure 5.3: Proof and verification time of zk-PoDI

We implemented the generator, prover, and verifier functions of zk-PoDI using the de-facto zk-SNARK library *libsnark*, and conducted experiments in an Ubuntu 22.04 workstation with Intel i9-13900KF CPU and 128GB DDR4 2133 DRAM. For each experiment, we repeated the prover and verifier functions 1000 times each and calculated the average proof and verification time.

The average proof time T_p and verification time T_v against Nx and Np are plotted in Figure 5.3. The T_p is shown in the upper plane with purple to yellow colors, and its contour is projected into x-y plane. The T_v is shown in bluish color at the bottom of the figure for comparison. From the plot, we can see that the T_v is nearly constant at approximately 3 ms, but the T_p varies from 8 to 72 ms. The T_p has a positive relationship with Nx and Np respectively, and increasing Nx has a similar effect of increasing $log_2(Np)$. This feature gives zk-PoDI the ability to freely adjust the ratio of T_p/T_v, which is considered convenient in designing the actual applications.

Also, with high enough T_p/T_v, zk-PoDI can be treated as a protocol with an embedded proof of work (PoW), which can then mitigate various attacks including denial of service (DoS) or brute force attacks, similar to the key derivation function (KDF).

While in practice, the number of surrounding vehicles is usually greater than one. A possible optimization could be done by proving the distinctiveness of several adjacent vehicles at a time. The corresponding DIC is as follows and the modification to the circuit is obvious:

$$\begin{cases} \mathcal{F}(\vec{S}) = 0 \\ \mathcal{G}(\vec{S}) \neq 0 \\ \mathcal{H}(\vec{S}) \neq 0 \\ \cdots \end{cases} \tag{5.15}$$

5.4.2 Threat Analysis

From the ZKP properties of the protocol, we directly know that a malicious prover cannot make the verifier convinced. However, it still may generate many false proofs to exhaust the verifier's computation power, which is known as DoS attack. Nevertheless, a DoS attack is still considered non-profitable. The most computationally heavy process is the VF and the verifier only evaluates the VF after the pseudonyms are successfully verified. This means the malicious vehicle must "honestly" generate the proofs using the collected real pseudonyms. From the performance analysis, we can confirm that the cost of generating a proof is much higher than verifying it. All those factors combined make the DoS attack meaningless.

Potential attacks based on the malleability of equations, including chosen plaintext attack, chosen identity attack, etc. are not possible because the coefficients of equations are directly generated from the pseudonym and the constant term is signed in conjunction with the pseudonym by the LEA.

Another potential attack is, since the pseudonyms (thus also equations) may be preloaded into the vehicles, an attacker may attempt to find solutions other than S in an offline fashion, with GPUs or ASICS. A possible mitigation is to quantize the difficulty of finding solutions and select an equation to make the expectation solving time much longer than the lifetime of a pseudonym.

5.5 Summary

In this chapter, we proposed the cryptographic protocol zero-knowledge Proof of Distinct Identity (zk-PoDI), which enables the vehicles to prove that it

is not the owner of another known pseudonym in the local area. Such a mechanism solved the paradox: it prevents the Sybil attack while maintaining the unlinkability of the pseudonym.

According to a recent survey [HIZo22], zk-PoDI fully fulfills all the requirements regarding a practical Sybil-resistance pseudonym system. zk-PoDI features zero latency, moderate computation overhead, and negligible communication cost. More importantly, it is designed to work independently and statelessly. It does not rely on a particular design of the underlying pseudonym system, does not rely on extra knowledge about the environment, the historical behavior of other vehicles, or any sort of assistance from the infrastructure. These features make the zk-PoDI very distinctive compared to the existing works and we believe that it is not far from being integrated into the C-ITS system.

However, we admit that the evaluation in a realistic environment is necessary, to verify the protocol itself, and its overall impact on the system. Therefore, in the future, we will implement the zk-PoDI in the Flowsim [TJN+23], with multiple pseudonym implementations and integration with different C-ITS applications.

Chapter 6

Conclusion

Ensuring no one is lying is always hard, and C-ITS is not an exception. In this dissertation, we made several attempts to give the vehicles confidence to tell the truth, without the risk of compromising their privacy; also, we give other vehicles mathematic tools to believe and respect that truth.

Firstly, in chapter 3, we presented Flowsim, a novel simulator that offers a modular and extensible architecture for evaluating CAV behavior and data flow in city-scale environments. Flowsim enables the analysis of various aspects of CAV performance, such as cybersecurity, efficiency, and latency, by providing a customizable and high-performance simulation platform. Flowsim is written in pure Python code and is available as an open-source project on GitHub, aiming to foster collaboration and innovation within the research community.

Using Flowsim, we are able to investigate data authenticity issues in cooperative perception. In chapter 4, we presented the zero-knowledge Proof of Traffic (zk-PoT.) Directed by our fact-based approach, zk-PoT enables vehicles to prove and cross-verify the existence of other vehicles they observed, without revealing any information about them. Zk-PoT can enhance the security, efficiency, and verification latency of CP, while preserving the privacy of the target vehicles. Zk-PoT can be easily integrated with existing CP standards, such as ISO and ETSI, and can also be used as a bootstrap method for trust establishment. We evaluate the performance of zk-PoT through simulation experiments in a realistic urban scenario and show that it achieves a high cross-verification ratio, low verification delay, and low bandwidth overhead. We also analyze the security and privacy properties of zk-PoT and discuss the possible countermeasures against various attacks.

The zk-PoT is especially vulnerable to the Sybil attack. Thus, in chapter 3, we presented a novel cryptographic protocol called zero-knowledge Proof of Distinct Identity (zk-PoDI), which enables a vehicle to prove that it

is not the owner of another pseudonym in the local area, without revealing its real identity or breaking the unlinkability of the pseudonyms. zk-PoDI is based on the Diophantine equation and zk-SNARK and can be integrated into most pseudonym systems as an extension. We show that zk-PoDI fulfills all the requirements of a practical Sybil-resistance pseudonym system, and has zero latency, moderate computation overhead, and negligible communication cost.

Admittedly, the problems we have solved so far are relatively straightforward. As our research advances, we will require more mathematical insights and a deeper comprehension of the challenges, and there is no guarantee of how far we can proceed in this direction. Nevertheless, we have confidence in the efficacy of fact-based approaches and zero-knowledge proofs. In the future, we will investigate more security and privacy issues of C-ITS, and endeavor to address them with fact-based techniques. We believe that trust will eventually be established, if everyone is obliged to tell the truth.

Bibliography

[AAAMA20] Muhammed Samir Al-Ali, Hasan Abbas Al-Mohammed, and Mahdi Alkaeed. Reputation based traffic event validation and vehicle authentication using blockchain technology. In *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*, pages 451–456, February 2020.

[AJN+23] Yu Asabe, Ehsan Javanmardi, Jin Nakazato, Manabu Tsukada, and Hiroshi Esaki. AutowareV2X: Reliable V2X communication and collective perception for autonomous driving. In *The 2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, Florence, Italy, 2023. published.

[AK23] Yasir Ali and Habib Ullah Khan. A survey on harnessing the applications of mobile computing in healthcare during the COVID-19 pandemic: Challenges and solutions. *Comput Netw*, 224:109605, April 2023.

[AL20] Ikram Ali and Fagen Li. An efficient conditional privacy-preserving authentication scheme for Vehicle-To-Infrastructure communication in VANETs. *Vehicular Communications*, 22:100228, April 2020.

[AMH23] Mahdi Akil, Leonardo Martucci, and Jaap-Henk Hoepman. Non-Interactive Privacy-Preserving Sybil-Free authentication scheme in VANETs. In *Symposium on Vehicles Security and Privacy (VehicleSec) 2023. 27 February, 2023, San Diego, CA, USA*. Internet Society, 2023.

[BDSMP91] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Noninteractive zero-knowledge. *SIAM Journal on Computing*, 20(6):1084–1118, 1991.

[BNM+22] Mohamed Baza, Mahmoud Nabil, Mohamed M E A Mahmoud, Niclas Bewermeier, Kemal Fidan, Waleed Alasmary, and Mohamed Abdallah. Detecting sybil attacks using proofs of work and location in VANETs. *IEEE Trans. Dependable Secure Comput.*, 19(1):39–53, January 2022.

[Bro09] Brown. Sec 1: Elliptic curve cryptography. *Certicom Research*, 2009.

[BSCG+14] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474, May 2014.

[CFE15] Lara Codeca, Raphael Frank, and Thomas Engel. Luxembourg SUMO traffic (LuST) scenario: 24 hours of mobility for vehicular networking research. In *2015 IEEE Vehicular Networking Conference (VNC)*, pages 1–8, December 2015.

[CMTSHG23] Raúl Casanova-Marqués, Joaquín Torres-Sospedra, Jan Hajny, and Michael Gould. Maximizing privacy and security of collaborative indoor positioning using zero-knowledge proofs. *Internet of Things*, 22:100801, 2023.

[CSA+21] Junaid Ahsenali Chaudhry, Kashif Saleem, Mamoun Alazab, Hafiz Maher Ali Zeeshan, Jalal Al-Muhtadi, and Joel J P C Rodrigues. Data security through Zero-Knowledge proof and statistical fingerprinting in Vehicle-to-Healthcare everything (V2HX) communications. *IEEE Trans. Intell. Transp. Syst.*, 22(6):3869–3879, June 2021.

[CZSL19] Pedro Cirne, André Zúquete, Susana Sargento, and Miguel Luís. The impact of ECDSA in a VANET routing service: Insights from real data traces. *Ad Hoc Networks*, 90:101747, July 2019.

[dB08] Mark de Berg. *Computational Geometry: Algorithms and Applications*. Springer Science & Business Media, March 2008.

[Doc] Sumo Documentation. Why vehicles are teleporting. `https://sumo.dlr.de/docs/Simulation/Why_Vehicles_are_teleporting.html`. Accessed: 2023-6-8.

[Dou02] John R Douceur. The sybil attack. In *Peer-to-Peer Systems*, pages 251–260. Springer Berlin Heidelberg, 2002.

[DRC+17] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 1–16. PMLR, 2017.

[EAM22] Secil Ercan, Marwane Ayaida, and Nadhir Messai. Misbehavior detection for position falsification attacks in VANETs using machine learning. *IEEE Access*, 10:1893–1904, 2022.

[ETS15] ETSI. Intelligent transport systems (ITS); security; security header and certificate formats. Technical Report TS 103-097, 2015.

[ETS19] ETSI. Intelligent transport system (ITS); vehicular communications. basic set of applications; analysis of the collective perception service (CPS). Technical Report 103 562, 2019.

[ETS20] ETSI. Intelligent transport systems (ITS); vehicular communications; GeoNetworking; part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; sub-part 1: Media-Independent functionality. Technical Report TS 102 636-4-1, 2020.

[ETS22] ETSI. Intelligent transport systems (ITS); security; trust and privacy management; release 2. Technical Report TS 102-941, 2022.

[ETS23a] ETSI. Intelligent transport system (ITS); vehicular communications; basic set of applications; collective perception service; release 2. Technical Report TS 103-562, 2023.

[ETS23b] ETSI. Intelligent transport systems (ITS); vehicular communications; basic set of applications; collective perception service; release 2. Technical Report TS 103 324, 2023.

[FFB16] S Friebe, M Florian, and I Baumgart. Decentralized and sybil-resistant pseudonym registration using social graphs. *2016 14th Annual Conference*, 2016.

[FWB15] M Florian, J Walter, and I Baumgart. Sybil-resistant pseudonymization and pseudonym change without trusted third parties. *Proceedings of the 14th ACM Workshop*, 2015.

[GGS04] Philippe Golle, Dan Greene, and Jessica Staddon. Detecting and correcting malicious data in VANETs. In *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, VANET '04, pages 29–37, New York, NY, USA, October 2004. Association for Computing Machinery.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, February 1989.

[Gro16] Jens Groth. On the size of Pairing-Based non-interactive arguments. In *Advances in Cryptology – EUROCRYPT 2016*, pages 305–326. Springer Berlin Heidelberg, 2016.

[GVG+10] Mainak Ghosh, Anitha Varghese, Arobinda Gupta, Arzad A Kherani, and Skanda N Muthaiah. Detecting misbehaviors in VANET with integrated root-cause analysis. *Ad Hoc Networks*, 8(7):778–790, September 2010.

[HEGK20] Shimaa A Abdel Hakeem, Mohamed A Abd El-Gawad, and Hyungwon Kim. Comparative experiments of V2X security protocol based on hash chain cryptography. *Sensors*, 20(19), October 2020.

[HIZo22] B Hammi, Y M Idir, S Zeadally, and others. Is it really easy to detect sybil attacks in C-ITS environments: a position paper. *IEEE Transactions on*, 2022.

[HLZ21] Rasheed Hussain, Jooyoung Lee, and Sherali Zeadally. Trust in VANET: A survey of current solutions and future research opportunities. *IEEE Trans. Intell. Transp. Syst.*, 22(5):2553–2571, May 2021.

[HMP22] Badis Hammi, Jean-Philippe Monteuuis, and Jonathan Petit. PKIs in C-ITS: Security functions, architectures and projects: A survey. *Vehicular Communications*, 38:100531, December 2022.

[HSBL17] Hamssa Hasrouny, Abed Ellatif Samhat, Carole Bassil, and Anis Laouiti. VANet security challenges and solutions: A survey. *Vehicular Communications*, 7:7–20, January 2017.

[Int20] International Standard Organization. Intelligent transport systems — station and communication architecture. Technical Report ISO 21217:2020, December 2020.

[Iso22] Iso. Information security, cybersecurity and privacy protection – evaluation criteria for IT security – part 2: Security

functional components. Technical Report 15408-2, International Organization for Standardization, Geneva, Switzerland, 2022.

[JMV01] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ECDSA). *Int. J. Inf. Secur.*, 1(1):36–63, August 2001.

[KKSS19] Rashmi Kushwah, Ayushi Kulshreshtha, Krishnapal Singh, and Shivanshu Sharma. ECDSA for data origin authentication and vehicle security in VANET. In *2019 Twelfth International Conference on Contemporary Computing (IC3)*, pages 1–5, August 2019.

[KLH23] Edy Kristianto, Po-Ching Lin, and Ren-Hung Hwang. Misbehavior detection system with semi-supervised federated learning. *Vehicular Communications*, 41:100597, June 2023.

[KR17] B Kaliski and A Rusch. PKCS #5: Password-based cryptography specification version 2.1. Technical Report rfc8018, January 2017.

[LAB21] Faiza Loukil, Mourad Abed, and Khouloud Boukadi. Blockchain adoption in education: a systematic literature review. *Educ. Inf. Technol.*, 26(5):5779–5797, September 2021.

[LBBW+18] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wiessner. Microscopic traffic simulation using SUMO. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2575–2582, November 2018.

[LMGN23] Wanxin Li, Collin Meese, Hao Guo, and Mark Nejad. Aggregated Zero-Knowledge proof and Blockchain-Empowered authentication for autonomous truck platooning. *IEEE Trans. Intell. Transp. Syst.*, 24(9):9309–9323, September 2023.

[LSL+21] J Li, Z Song, Y Li, C Cao, and Y He. Trajectory as an identity: Privacy-preserving and sybil-resistant authentication for internet of vehicles. *Security and Communication*, 2021.

[LSS+08] Tim Leinmuller, Robert K Schmidt, Elmar Schoch, Albert Held, and Gunter Schafer. Modeling roadside attacker behavior in VANETs. In *2008 IEEE Globecom Workshops*, pages 1–10, November 2008.

[LT07] Nai-Wei Lo and Hsiao-Chien Tsai. Illusion attack on VANET applications - a message plausibility problem. In *2007 IEEE Globecom Workshops*, pages 1–8, November 2007.

[MGGR13] Ian Miers, Christina Garman, Matthew Green, and Aviel D Rubin. Zerocoin: Anonymous distributed E-Cash from bitcoin. In *2013 IEEE Symposium on Security and Privacy*, pages 397–411, May 2013.

[MK22] Jordan McEntyre and Billy Kihei. Zero-Knowledge proof for enabling privacy preserving electronic toll collection with Vehicle-to-Everything communications. In *2022 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–6, January 2022.

[MMT+23] Hidetaka Masuda, Oussama El Marai, Manabu Tsukada, Tarik Taleb, and Hiroshi Esaki. Feature-Based vehicle identification framework for optimization of collective perception messages in vehicular networks. *IEEE Trans. Veh. Technol.*, 72(2):2120–2129, February 2023.

[MMZo21] D Maram, H Malvai, F Zhang, and others. Candid: Cando decentralized identity with legacy compatibility, sybil-resistance, and accountability. *IEEE Symposium on ...*, 2021.

[MPA+22] Shanmukha Makani, Rachitha Pittala, Eitaa Alsayed, Moayad Aloqaily, and Yaser Jararweh. A survey of blockchain applications in sustainable and smart cities. *Cluster Comput.*, 25(6):3915–3936, December 2022.

[MYS+20] Pravin Mundhe, Vijay Kumar Yadav, Abhishek Singh, Shekhar Verma, and S Venkatesan. Ring signature-based conditional privacy-preserving authentication in VANETs. *Wirel. Pers. Commun.*, 114(1):853–881, September 2020.

[MZTC11] Umar Farooq Minhas, Jie Zhang, Thomas T Tran, and Robin Cohen. A multifaceted approach to modeling agent trust for effective communication in the application of mobile ad hoc

vehicular networks. *IEEE Trans. Syst. Man Cybern. C Appl. Rev.*, 41:407–420, May 2011.

[Nak08] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer electronic cash system, December 2008.

[Pet09] Jonathan Petit. Analysis of ECDSA authentication processing in VANETs. In *2009 3rd International Conference on New Technologies, Mobility and Security*, pages 1–5, December 2009.

[PH10] Andreas Pfitzmann and Marit Hansen. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management, 2010.

[PJ16] C Percival and S Josefsson. The scrypt Password-Based key derivation function. Technical Report rfc7914, August 2016.

[PLH+21] Kai Peng, Meijun Li, Haojun Huang, Chen Wang, Shaohua Wan, and Kim-Kwang Raymond Choo. Security challenges and opportunities for smart contracts in internet of things: A survey. *IEEE Internet Things J.*, 8(15):12004–12020, August 2021.

[PSFM20] Francesco Pollicino, Dario Stabili, Luca Ferretti, and Mirco Marchetti. An experimental analysis of ECQV implicit certificates performance in VANETs. In *2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*, pages 1–6, November 2020.

[RH10] George F Riley and Thomas R Henderson. The ns-3 network simulator. In Klaus Wehrle, Mesut Güneş, and James Gross, editors, *Modeling and Tools for Network Simulation*, pages 15–34. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[RMHL20] Amar A Rasheed, Rabi N Mahapatra, and Felix G Hamza-Lup. Adaptive Group-Based zero knowledge Proof-Authentication protocol in vehicular ad hoc networks. *IEEE Trans. Intell. Transp. Syst.*, 21(2):867–881, February 2020.

[ROG19] Raphael Riebl, Christina Obermaier, and Hendrik-Jörn Günther. Artery: Large scale simulation environment for ITS applications. In Antonio Virdis and Michael Kirsche, editors,

Recent Advances in Network Simulation: The OMNeT++ Environment and its Ecosystem, pages 365–406. Springer International Publishing, Cham, 2019.

[RPGH08] M Raya, P Papadimitratos, V D Gligor, and J-P Hubaux. On Data-Centric trust establishment in ephemeral ad hoc networks. In *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, pages 1238–1246, April 2008.

[RRST21] Abderahman Rejeb, Karim Rejeb, Steve Simske, and Horst Treiblmaier. Blockchain technologies in logistics and supply chain management: A bibliometric review. *Logistics*, 5(4):72, October 2021.

[RSF+19] Joao Rufino, Luis Silva, Bruno Fernandes, Joao Almeida, and Joaquim Ferreira. Overhead of V2X secured messages: An analysis. In *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, pages 1–5, April 2019.

[RST+20] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, Eugene Agafonov, Tae Hyung Kim, Eric Sterner, Keunhae Ushiroda, Michael Reyes, Dmitry Zelenkovsky, and Seonman Kim. LGSVL simulator: A high fidelity simulator for autonomous driving. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. ieeexplore.ieee.org, September 2020.

[SGD11] C Sommer, R German, and F Dressler. Bidirectionally coupled network and road traffic simulation for improved IVC analysis. *IEEE Trans. Mob. Comput.*, 10(1):3–15, January 2011.

[SKVG+23] Roshan Sedar, Charalampos Kalalas, Francisco Vázquez-Gallego, Luis Alonso, and Jesus Alonso-Zarate. A comprehensive survey of V2X cybersecurity mechanisms and future research paths. *IEEE Open Journal of the Communications Society*, 4:325–391, 2023.

[Soc21] Society of Automotive Engineers. Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems. Technical Report J3016, 2021.

[SS19] 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, October 2019.

[Sta21] State Council of the People's Republic of China. Policy to promote autonomous driving. `https://english.www.gov.cn/policies/policywatch/202101/05/content_WS5ff39f9fc6d0f725769433b4.html`, January 2021.

[TB] Jan Trauernicht and Dr-Ing Norbert Bißmeyer. Deterministic sybil attack exclusion in Cooperative-Intelligent transportation systems. `https://scholar.archive.org/work/gfoctp3hvndo3fcihabazgefyi/access/wayback/https://hss-opus.ub.ruhr-uni-bochum.de/opus4/frontdoor/deliver/index/docId/6655/file/Kapitel4.pdf`. Accessed: 2023-11-14.

[TJL+23] Ye Tao, Yuze Jiang, Pengfei Lin, Manabu Tsukada, and Hiroshi Esaki. Zk-PoT: Zero-knowledge proof of traffic for privacy enabled cooperative perception. In *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*, pages 261–268. IEEE, January 2023.

[TJN+23] Ye Tao, Ehsan Javanmardi, Jin Nakazato, Manabu Tsukada, and Hiroshi Esaki. Flowsim: A modular simulation platform for microscopic behavior analysis of City-Scale connected autonomous vehicles. In *The 26th edition of the IEEE International Conference on Intelligent Transportation Systems (ITSC 2023)*, Bilbao, Bizkaia, Spain, 2023. published.

[Var10] Andras Varga. OMNeT++. In Klaus Wehrle, Mesut Güneş, and James Gross, editors, *Modeling and Tools for Network Simulation*, pages 35–59. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[VS22] Sanjeev Verma and Ashutosh Sheel. Blockchain for government organizations: past, present and future. *Journal of Global Operations and Strategic Sourcing*, 15(3):406–430, January 2022.

[WHM+23] Zhangquan Wang, Jiaxuan Huang, Kelei Miao, Xiaowen Lv, Yourong Chen, Bing Su, Liyuan Liu, and Meng Han. Lightweight zero-knowledge authentication scheme for iot embedded devices. *Computer Networks*, 236:110021, 2023.

[XGH+21] Runsheng Xu, Yi Guo, Xu Han, Xin Xia, Hao Xiang, and Jiaqi Ma. OpenCDA: An open cooperative driving automation framework integrated with co-simulation. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, September 2021.

[XLJM22] Ning Xi, Weihui Li, Lv Jing, and Jianfeng Ma. Zama: A zkp-based anonymous mutual authentication scheme for the iov. *IEEE Internet of Things Journal*, 9(22):22903–22913, 2022.

[YCT+19] 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.

[ZF18] Jithin Zacharias and Sibylle Fröschle. Misbehavior detection system in VANETs using local traffic density. In *2018 IEEE Vehicular Networking Conference (VNC)*, pages 1–4, December 2018.

Publications

Publications Related to the Thesis

[1] **Ye Tao**, Ehsan Javanmardi, Pengfei Lin, Yuze Jiang, Jin Nakazato, Manabu Tsukada, Hiroshi Esaki. "Zero-Knowledge Proof of Traffic: A Deterministic and Privacy-Preserving Cross Verification Scheme for Cooperative Perception Data." IEEE Access.

[2] **Ye Tao**, Yuze Jiang, Pengfei Lin, Manabu Tsukada, Hiroshi Esaki. "Zk-PoT: Zero-Knowledge Proof of Traffic for Privacy Enabled Cooperative Perception." 2023 IEEE 20th Consumer Communications & Networking Conference (IEEE CCNC 2023).

[3] **Ye Tao**, Ehsan Javanmardi, Jin Nakazato, Manabu Tsukada, Hiroshi Esaki. "Flowsim: A Modular Simulation Platform for Microscopic Behavior Analysis of City-Scale Connected Autonomous Vehicles." 2023 IEEE 26th International Conference on Intelligent Transportation Systems (IEEE ITSC 2023).

[4] **Ye Tao**, Hongyi Wu, Ehsan Javanmardi, Manabu Tsukada, Hiroshi Esaki. "Zero-Knowledge Proof of Distinct Identity: a Local Sybil-resistant Pseudonym Extension for ITS." 2024 IEEE 15th Vehicular Networking Conference (VNC2024). Under submission.

Other Publications

[1] **Ye Tao**, Xin Li, Manabu Tsukada, Hiroshi Esaki. "Reliable Overlay Networking on ETSI GeoNetworking Standards." International Journal of Intelligent Transportation Systems Research 16 (2): 98–111, 2018.

[2] **Ye Tao**, Manabu Tsukada, Hiroshi Esaki. "Positioning and Perception in Cooperative ITS Application Simulator." The Sixth International Conference on Advances in Vehicular Systems, Technologies and Applications (VEHICULAR 2017).

[3] **Ye Tao**, Xin Li, Manabu Tsukada, Hiroshi Esaki. "DUPE: Duplicated Unicast Packet Encapsulation in Position-Based Routing VANET." 2016 9th IFIP Wireless and Mobile Networking Conference (WMNC 2016).

[4] **Ye Tao**, Manabu Tsukada, Xin Li, Masayuki Kakiuchi, Hiroshi Esaki. "Reproducing and Extending Real Testbed Evaluation of GeoNetworking Implementation in Simulated Networks." The 10th International Conference on Future Internet (ACM CFI 2015).

[5] Pengfei Lin, Ehsan Javanmardi, **Ye Tao**, Vishal Chauhan, Jin Nakazato, Manabu Tsukada. "Time-To-Collision-Aware Lane-Change Strategy Based on Potential Field and Cubic Polynomial for Autonomous Vehicles." 2023 IEEE Intelligent Vehicles Symposium (IEEE IV 2023).