Secure and Privacy-preserving Data Aggregation in Internet of Vehicles

by

Rui Liu
B.Sc., China University of Geosciences, 2018

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Computer Science

Secure and Privacy-preserving Data Aggregation in Internet of Vehicles

by

Rui Liu
B.Sc., China University of Geosciences, 2018

Supervisory Committee

Dr. Jianping Pan, Supervisor
(Department of Computer Science)

Dr. Bruce Kapron, Departmental Member
(Department of Computer Science)

Dr. Issa Traore, Outside Member
(Department of Electrical and Computer Engineering)

ABSTRACT

In Internet of Vehicles (IoV), crucial data is aggregated to support the applications for automatic driving, intelligent transportation and smart cities. It is crucial to carefully address certain challenges in this process, particularly regarding security and privacy.

In this dissertation, we first target a representative IoV data aggregation scenario, fine-grained air quality monitoring. The major challenges we focus on include: a) the sensory data provided by vehicles usually vary in quality; b) there is a significant difference in traffic volumes of streets or blocks, which leads to a data sparsity problem; and c) the original sensory data, vehicle identities, and trajectories face risks of exposure. To address these issues, we propose a truth discovery algorithm incorporating multiple correlations, and extend it to a privacy-preserving framework, EAirQ.

EAirQ relies on a traditional end-to-end data aggregation architecture. Designing a new architecture specifically for vehicular networks may hold significant value. Thus, we introduce a privacy-preserving two-layered architecture with vehicle clusters. Instead of focusing on a specific application, we present how this architecture can be well adopted in a general distributed machine learning scenario. We named this part of the work CRS. CRS not only protects the local data, the identities and trajectories of vehicles, but also ensures the accuracy of aggregated learning models by handling packet loss in the application layer.

We further work on eliminating the limitations of the proposed two-layered architecture in the following three aspects: a) to provide fast and easy verification of messages within a cluster; b) to preserve vehicle privacy without adopting the pseudonym technique; c) to consider the adversarial behaviors of vehicles and enhance the security. Our solution introduces a novel concept, data approval, based on the Schnorr signature scheme. This part of the work, named SADA, meets more security requirements and is lightweight for vehicles.

In addition to exploring new solutions to preserve the privacy of vehicle identities and trajectories, we also pay attention to the latest industry standards. This part of the work focuses on tackling the challenge of certificate provisioning in the latest solution to satisfy the anonymous communication requirement in IoV. We propose a non-interactive approach, named NOINS, empowering vehicles to generate short-term key pairs and anonymous implicit certificates on their side. This new paradigm introduces the possibilities for many extensions and applications.

Table of Contents

Supervisory Committee **ii**

Abstract **iii**

Table of Contents **iv**

List of Tables **viii**

List of Figures **ix**

Acknowledgements **x**

1 Introduction **1**

1.1 Background 1

1.2 Research Motivation 4

1.3 Research Issues and Contributions 6

2 Lightweight Privacy-preserving Truth Discovery for Vehicular Air Quality Monitoring **11**

2.1 Introduction 11

2.2 Related Work 14

2.3 Cryptography Tools 15

2.4 Problem Definition 16

2.5 Framework Design 18

2.5.1 ST algorithm 18

2.5.2 SST algorithm 21

2.5.3 Masking mechanism 22

2.5.4 Perturbation mechanism 25

2.5.5 EAirQ framework 26

2.6 Performance Evaluation 29
2.6.1 Truth discovery performance 29
2.6.2 Computation cost 36
2.6.3 Communication cost 37
2.6.4 Privacy analysis 38
2.7 Discussion 40
2.8 Conclusion 43

3 CRS: Two-layered Privacy-preserving Distributed Machine Learning 44
3.1 Introduction 46
3.2 Related Work 48
3.2.1 Privacy preservation in machine learning 48
3.2.2 Clustering techniques in IoV 49
3.3 Cryptographic Tools and Adopted Techniques 50
3.4 Problem Definition and Threat Model 52
3.4.1 Problem definition 52
3.4.2 Threat model 53
3.5 The CRS Framework 54
3.5.1 Cluster generation and setup 54
3.5.2 Reliable and secure data aggregation 57
3.5.3 Two-layered secure learning framework 58
3.6 Application and Prospects 60
3.6.1 An application instance 60
3.6.2 More use cases and prospects 64
3.7 Performance Evaluation 65
3.7.1 Simulation setup 65
3.7.2 Packet loss and recovery 67
3.7.3 Communication cost 69
3.7.4 Computation cost 71
3.7.5 Evaluation with the large w_i 72
3.7.6 Security and privacy analysis 72
3.7.7 Summary and comparison 74
3.8 Discussion and Future Work 75
3.9 Conclusion 78

4 Schnorr Approval-based Secure and Privacy-preserving IoV Data Aggregation 80
4.1 Introduction . . . 81
4.2 Related Work . . . 84
4.3 Preliminaries . . . 85
4.4 System Model and Threat Model . . . 87
4.4.1 System model . . . 87
4.4.2 Threat model . . . 88
4.5 Framework Overview . . . 89
4.6 Framework Design . . . 92
4.6.1 Recoverable masking protocol . . . 92
4.6.2 Data approval generation . . . 94
4.6.3 Data approval pre-checking . . . 95
4.6.4 Regional data uploading . . . 97
4.6.5 Cluster key audit and bad head identification . . . 98
4.7 Performance Evaluation . . . 98
4.7.1 Computation cost . . . 99
4.7.2 Communication cost . . . 103
4.8 Security Analysis . . . 104
4.8.1 Data leakage attack . . . 105
4.8.2 Identity leakage attack and trajectory tracking attack . . . 106
4.8.3 Fake data injection attack . . . 107
4.8.4 Input invalidation attack . . . 108
4.9 Conclusion . . . 108

5 Flexible Non-interactive Short-term Implicit Certificate Generation for IoV 109
5.1 Introduction . . . 111
5.2 Related Work . . . 113
5.2.1 SCMS . . . 113
5.2.2 Sanitizable signature . . . 113
5.2.3 Self-changing pseudonyms . . . 114
5.3 System Model and Threat Model . . . 115
5.3.1 System model . . . 115
5.3.2 Threat model . . . 116

5.4 NOINS Design 117
5.4.1 CA-issued certificate generation 118
5.4.2 Short-term certificate generation 120
5.4.3 Short-term certificate using in V2X communications 122
5.5 Performance Evaluation 123
5.6 Security Analysis 125
5.7 Discussion 129
5.8 Conclusion 131

6 Conclusion **132**

Bibliography **134**

Appendix A Proof of Theorem 4.1 **150**

List of Tables

Table 2.1 Notations in Chapter 2 12
Table 2.2 Parameter settings for comparison 34
Table 2.3 Comparison on the computation cost 36

Table 3.1 Notations in Chapter 3 45
Table 3.2 Cryptographic operations and execution time [76] in Chapter 3 . 68
Table 3.3 Comparison on security, privacy and accuracy 74
Table 3.4 Comparison on the critical communication cost of vehicles . . . 75
Table 3.5 Comparison on the critical computation cost of vehicles ($n = 20$) 76

Table 4.1 Notations in Chapter 4 81
Table 4.2 Cryptographic operations and execution time [76] in Chapter 4 . 99
Table 4.3 Comparison on the computation cost of secure data aggregation 100
Table 4.4 Comparison on the cost of message verification in VANETs . . . 102
Table 4.5 Comparison with CRS on message verification and identity privacy preservation 102
Table 4.6 Computation time of SADA 103
Table 4.7 Comparison on security and privacy protection 105

Table 5.1 Notations in Chapter 5 110
Table 5.2 Certificate generation and provisioning process 119
Table 5.3 Computation cost for n_c certificates (in milliseconds) 124
Table 5.4 Communication time required for obtaining n_c certificates (in seconds) 125
Table 5.5 Total communication time for obtaining and using n_c certificates (in seconds) 125

List of Figures

Figure 2.1 EAirQ framework processes 27
Figure 2.2 Zipf and Laplace distributions 29
Figure 2.3 Observation values generation 31
Figure 2.4 Performance of EAirQ with good sources 33
Figure 2.5 RMSE differences between EAirQ and AirQ with different perturbation parameters 34
Figure 2.6 Performance of EAirQ with 15% bad sources 35

Figure 3.1 Architecture of CRS 58
Figure 3.2 CRS Framework 59
Figure 3.3 CRS application example 63
Figure 3.4 Vehicle position and mobility in the emulated scenario 66
Figure 3.5 Simulation results of CRS with the emulated highway scenario . 69
Figure 3.6 Simulation results of CRS with the realistic scenario 69

Figure 4.1 System model of SADA 87
Figure 4.2 Protocol of SADA 90
Figure 4.3 An example of the maintained trees with $n_{\mathrm{v}} = 10$ 96
Figure 4.4 Comparison of the bad sub-approval identification 101

Figure 5.1 System model of NOINS 116

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to my supervisor, Dr. Jianping Pan, for mentoring, support, encouragement, and patience. I sincerely appreciate all the professors who provided invaluable guidance throughout this long journey, especially Dr. Lin Cai and Dr. Jun Song. The ensuring unwavering support from my family and friends is extremely meaningful to me. Thanks for all the smiles, hugs, kisses, and companionship, which meant the world to me. I am grateful for everyone who appears in my life.

I know it is a long and challenging journey but I always remind myself: life is tough but so you are. It is the little but beautiful things — a rainbow, a morning hi, a stranger's smile, a dog passing by — that brighten my days and bring courage for me to overcome all the difficulties. I wish you, who are reading this dissertation, full of happiness in your life as well.

Rui Liu

Chapter 1

Introduction

1.1 Background

Vehicular ad-hoc networks (VANETs) are mobile ad-hoc networks of vehicles and infrastructures. In VANETs, vehicles are equipped with on-board units (OBUs) which handle computation and communication tasks. These vehicles exchange messages with both other vehicles and infrastructures, referred to as V2V (vehicle-to-vehicle) communication mode and V2I (vehicle-to-infrastructure) communication mode, respectively. One common type of infrastructure is often referred to as Road-Side Units (RSUs). RSUs are fixed units positioned along roadways to monitor vehicles, gather information and provide Internet access. They typically have a much wider communication range than vehicles.

Communication quality and cost play a vital role in VANETs. Some researchers work on the access standards and communication technologies to support a larger transmission range, a higher data rate, and an enhanced reliability [68, 129]. Some other works focus on designing suitable routing protocols for VANETs [61, 33]. These works are challenging on account of the high-speed movement of vehicles which leads to rapidly changed network topologies and unstable connectivity. In addition, security and privacy requirements should be considered carefully when implementing a VANET system. Related topics, such as physical security, trust management, privacy preservation and advanced attacks, have been studied [60, 114, 39, 58, 4].

The **Internet of Vehicles** (IoV) is a promising evolution from Internet of Things (IoT) and VANETs. The name of IoV refers to that a wide range of VANET objects (including vehicles and infrastructures) and potential application servers are

connected with each other and the Internet. In IoV, in addition to OBUs, vehicles are outfitted with specialized equipments, such as sensors and cameras. With these equipments, vehicles are able to contribute a wealth of valuable data to the vehicular network or designated servers: not only the data related to a vehicle itself such as the speed, but also the data of environments. The exponential growth of data volume in IoV and the ongoing technological advancements give rise to many potential scenarios and applications. We list a few examples as follows:

Road safety: instead of requiring the driver to manually determine the safe following distance and driving speed, it is possible to generate a real-time smart recommendation with a machine learning model trained on the data of vehicle speeds, accelerations, road conditions, and weather conditions.

Urgent situation response: malfunctioning vehicles (e.g., a vehicle with a faulty brake) and emergency vehicles who need immediate right-of-way can broadcast their positions, speeds, and intended routes to nearby vehicles. It enables other vehicles to quickly react to emergency situations and clear the affected path.

Collision avoidance: vehicles can exchange critical information regarding their speeds, positions, accelerations and directions. Relying on this data, a collision avoidance system can provide an intelligent driving suggestion to avoid traffic accidents. In autonomous driving, this information plays an even more important role in making driving decisions.

Traffic management: the traffic status information, such as the number of vehicles, frequency of braking events, and vehicle speeds, can be gathered to monitor the traffic on the road, manage the signal lights, and avoid traffic congestions. It can help with optimizing traffic flow, improving road safety and alleviating human efforts in traffic management.

Parking assistance: by gathering real-time parking information and pricing details from both individual vehicles and parking facilities, a parking assistance system can help drivers quickly find available parking spaces.

Traffic prediction: the information on vehicle speeds, traffic congestion and travel patterns is valuable in predicting the traffic on roads. It can be realized with a machine learning model. Factors such as weather conditions and holidays may be taken into consideration as well.

Road condition and map updates: vehicles driving on the road can provide real-time updates about uneven road surfaces, slippery roads, road construction events, traffic accidents, traffic restrictions, changes of local maps, cleaning status of snow-covered roads and so on. Once these reports are verified, they can benefit all drivers and road maintenance teams, enhancing driving experience and road safety.

Environmental monitoring: vehicles equipped with sensors can collect data on air quality, weather conditions and noise levels of the passing-by streets. It enables real-time, fine-grained and localized environmental monitoring without the requirement of establishing a huge amount of monitoring stations.

In all the above applications, it is required to collect data from individual vehicles. This process is named *vehicular crowdsensing*. In IoV, it can be classified into two categories considering the entity responsible for gathering data:

- Vehicles: vehicles may exchange and collect data from neighboring vehicles. It can be achieved directly through V2V communication or with the assistance of RSUs. In this case, the analysis or decision is usually made directly by each individual vehicle. For example, in an emergency vehicle notification system, vehicles receive information about emergency vehicles from nearby vehicles or RSUs directly take immediate actions and clear the route.

- Servers: a server may acquire data from all local vehicles within a specific region or from all vehicles registered in the system. It can perform further operations for specific purposes. For example, in a localized weather monitoring system, the server collecting weather information from all individual vehicles can update the accurate local weather and assist drivers in adapting to changing weather conditions.

Data aggregation is a closely relative concept with data collection but it involves operations such as averaging, summing, counting, or other mathematical or logical processes to derive meaningful insights from the collected data. In this dissertation, we focus on averaging/summing the collected data with a central server in IoV. It can cover many concrete applications but also faces significant challenges. We describe the challenges that motivate our research in Section 1.2.

1.2 Research Motivation

While data aggregation in IoV holds significant value, it is crucial to carefully address certain challenges to ensure its practicality and encourage more vehicles to participate and contribute. In this section, we give a brief summary of the challenges that motivated our research.

Data privacy of vehicles: participating vehicles may be hesitant to upload their raw data due to concerns about privacy. For instance, the videos captured by dashcams may contain photos of the home and family members of the driver. Sometimes, even though drivers do not know what and how sensitive information can be revealed from their raw data, they are uncomfortable sharing their data as well. Thus, ensuring the confidentiality of the gathered data is important and necessary.

Considering the high mobility of vehicles, which leads to the rapid change in network topology, unreliable communication, relatively high delay, and short link lifetime, the traditional methods used in IoT may not be suitable for IoV. For example, homomorphic encryption introduces not only a high computation cost but also a high communication cost for vehicles. It also requires multiple communication rounds between the server and vehicles. Another popular low-cost method, differential privacy, unfortunately, reduces the accuracy of data. Finding an efficient solution for IoV data aggregation that ensures both security and usability is challenging, especially taking other challenges into consideration at the same time.

Identity privacy of vehicles: as participants in data aggregation tasks, vehicles may have a strong desire to remain anonymous. For example, in a task of driving behavior collection, participating drivers upload the driving data for rewards but may hope the data cannot be linked to their identities (e.g., an ID signed by a trusted authority) by others.

One typical solution is to use a pseudo-ID instead of the real ID. However, using a pseudonym does not mean the privacy of identity is totally preserved. The linkage of multiple data uploading events or communications with the same pseudonym also leaks the privacy of vehicles. Frequently changing pseudonyms usually means high computation, communication or storage costs to vehicles. Other unchanged values, such as public keys, should also be carefully considered.

Trajectory privacy of vehicles: nowadays, vehicles have become an almost indispensable part of most families. Different from participants in IoT which usually have fixed positions, vehicles in IoV are on the move. A rising concern is the exposure of trajectories, which may further reveal the locations of drivers' home and workplaces, travel preferences and daily routines. Both the uploaded data and the communication between vehicles and servers may expose the trajectories. As mentioned above, carefully using the pseudonym-based anonymous communication can solve the problem to some extent, as long as the pseudonyms are frequently changed and no values are linkable. However, considering the advanced data aggregation applications in IoT, the amount of vehicle-to-everything (V2X) communication will be increased drastically. A new solution is required to facilitate the higher demand for pseudonyms while reducing the communication cost of vehicles.

Physical attacks, such as identifying a vehicle by its color or license plate, or capturing the trajectories of vehicles with cameras on highways, also pose a risk to the privacy of vehicles. We believe there is a nonnegligible requirement for corresponding laws and regulations in real world to fully protect the identities and trajectories of vehicles but these are out of the scope of the dissertation. Thus, we do not consider these physical attacks.

Accuracy and reliability: the accuracy and reliability of the aggregation results have a great influence on the performance of the IoV applications with vehicular crowdsensing. They are affected by many factors, including the precision of the equipment used (typically sensors), the volume of the data, and the distribution of the data. Therefore, to meet the requirement of accuracy, it is necessary to address biased data and consider different situations in IoV. The requirement also leads to a constraint of system design: some privacy-preservation solutions, such as differential privacy, decrease the accuracy of data so should be avoided to some extent.

In addition, these applications involve a large number of participating vehicles but not all of them can be fully trusted. Thus, in case some fake data affect the reliability of the results, we also need to take misbehavior detection and trust management into consideration.

Misbehavior detection and trust management: in IoV, especially the scenarios

of crowdsensing, servers have limited control of participants, i.e., vehicles. Some vehicles may misbehave for interests. For instance, they might upload fake data which deviates from the real value. This behavior not only affects the reliability of the aggregation results but also obstructs the establishment of fair market competition. The privacy preservation of the individual data makes it more difficult to address the problem.

Uploading fake individual data is not the only misbehavior in IoV data aggregation. Other concrete misbehaviors are further considered in different problems and frameworks, for example, in certificate provisioning, corrupt vehicles may try to forge a certificate. Detecting such undesirable behaviors and managing the credibility of participants are necessary.

Communication security: to complete the data uploading tasks, vehicles need to communicate with RSUs, servers or each other. The confidentiality, integrity and authenticity of the messages sent should be guaranteed.

Workload of vehicles: although some technical solutions are necessary to tackle the above challenges of security and privacy, both the computation and communication costs of vehicles should be restricted to an acceptable range. The reasons are twofold: a) users are concerned most about the workload on their own sides; b) the computation and communication resources on vehicles are usually limited, which is a typical limitation in VANETs, compared with traditional wireless networks.

1.3 Research Issues and Contributions

Motivated by the challenges introduced in Section 1.2, we propose some research issues. The proposed issues and our corresponding contributions are as follows:

1. Air pollution has become an important health concern. The recent developments in vehicular networks and crowdsensing systems make it possible to monitor fine-grained air quality with vehicles. On account of the different precisions of onboard sensors and potential malicious behaviors of participants, sensory data usually vary in quality. Thus, truth discovery has been a crucial task which targets better utilizing the data. However, in urban cities, there is a

significant difference in traffic volumes of streets or blocks, which leads to a data sparsity challenge in truth discovery.

How can we tackle the challenge of data sparsity in truth discovery? In addition, how can we protect the vehicles' privacy without introducing excessive overhead (i.e., communication and computation costs) on vehicles in the process? Traditional solutions in crowdsensing [125, 120] cannot well adapt to fine-grained air quality monitoring considering the different demands and challenges. In Chapter 2, we target solving these problems in this concrete data aggregation scenario. We combine spatial and temporal correlations and propose two optimization-based truth discovery algorithms to solve the data sparsity problem and reduce the negative impact of data reuse. We further propose a privacy-preserving framework, EAirQ, based on the technique of masking, anonymous authentication and perturbation. The framework protects the observation values and the trajectories of vehicles. Different from the existing privacy-preserving methods based on cryptography [14, 119, 112, 54, 56], it is lightweight for vehicles.

This work has been published in the *16th International Conference on Mobility, Sensing and Networking* (MSN 2020) [63] and *Digital Communications and Networks* (DCN 2023) [64].

2. Nowadays, vehicles can provide many valuable data for analytical model building. Providing the data to a central server for model training leads to a high computation cost for the server and introduces a high communication cost for vehicles while potentially compromising vehicle privacy. Distributed machine learning (DML) has become a better choice because it eliminates the requirement of raw data collection and allows vehicles to train their data locally. However, the local training results aggregated by the server in DML may reveal information on the raw data [84]. It is still a challenge to further preserve the privacy while keeping both the computation and communication costs of vehicles acceptable.

   How to preserve the privacy of the local inputs and model weight vectors? How to protect the identities and trajectories of vehicles? How to make the DML framework more suitable for IoV? Different from the end-to-end architecture used in our first work presented in Chapter 2, we propose a specialized two-layered architecture for DML in VANETs, which has natural advantages: both the overhead of vehicles is reduced and the privacy is preserved. We present a

distributed machine learning framework, CRS, based on this architecture. In CRS, a proposed secure and reliable data aggregation protocol and a threshold homomorphic cryptography system work together to provide confidentiality and preserve privacy. Some related works [84, 41] solely relying on homomorphic encryption lead to high communication and computation costs to vehicles. Besides, different from existing works that add noise to the data [44, 42, 90], CRS does not reduce the aggregation accuracy. To the best of our knowledge, this is the first work that aims at designing a machine learning framework specifically for IoV, where the data privacy, identity privacy, trajectory privacy, packet loss, and communication and computation costs of vehicles are all considered. This work is introduced in Chapter 3.

This work has been published in the *IEEE Internet of Things Journal* (IoTJ 2024) [65].

3. The cluster-based two-layered architecture presented in our second work is specifically designed for IoV. It can be used for aggregating not only the local training results but also the individual sensing data, making it suitable for many data aggregation applications in IoV. Our third work is based on the proposed architecture but aims to overcome some limitations and provide advanced security protection. More specifically, we are attempting to answer the following questions while keeping the good properties of the previous work:

   How to verify messages within a vehicle cluster? Can we keep anonymous for vehicles but avoid the general solution in the existing works [115, 77, 57], where the management and update of pseudo-IDs bring high computation and communication costs to vehicles? What if a cluster head uploads a fake value? In Chapter 4, we propose a Schnorr approval-based data aggregation framework SADA. In SADA, we present a recoverable masking technique to preserve the privacy of sensory data, which enables a cluster to recover from an input invalidation attack. The new concept, data approval, can preserve the identity and trajectory privacy and defend against the fake data injection attack. SADA not only addresses the above problems, but also achieves the separation of liability among vehicles, which is a valuable advantage in the two-layered architecture.

   In Chapters 2 and 3, we analyze the security of the proposed frameworks regarding general goals. In Chapter 4, differently, we target more concrete security attacks, make more formal definitions, and formally prove the security of the

approval scheme in the Random Oracle Model.

This work has been submitted to the *IEEE Transactions on Vehicular Technology* [66].

4. A leading industry solution for secure and trusted communication in vehicular ad-hoc networks (VANETs) is the Security Credential Management System (SCMS) [19]. In SCMS, certificate authorities (CAs) generate and issue anonymous certificates to vehicles, which are used to preserve the privacy of vehicles. To facilitate the advanced data aggregation applications in IoV, vehicles are required to frequently change their anonymous certificates, which leads to high communication or storage costs. In addition, there is a lack of flexibility in the current design of SCMS. Some researchers have worked on improving SCMS in different aspects [79, 36, 95] but there is a lack of research on addressing these new challenges.

   In our third work, SADA, we tried to achieve the identity privacy preservation for vehicles by the two-layered architecture and the proposed Schnorr approval. Another question comes to our mind, if we follow the latest industry solution, SCMS, and adopt the anonymous credentials in VANETs, how can we further improve the efficiency while fully supporting the requirement of pseudonyms? Can we enable vehicles to have personalized certificate provisioning models? These problems are addressed in our fourth work, which is introduced in Chapter 5. In this work, vehicles are allowed to generate short-term certificates non-interactively, which avoids frequently establishing connectivity or downloading certificates from CAs or RSUs. They can personalize their certificate generation, e.g., change pseudonyms according to their driving habits and privacy requirements. Most of the similar works in self-changing the pseudonyms are not truly non-interactive [2, 121]. To the best of our knowledge, this is the first work applying (and further revising) sanitizable signatures in the non-interactive anonymous certificate generation problem and specifically designed for SCMS.

   This work is to be submitted to the *ACM Transactions on Privacy and Security* [62].

In this dissertation, we mainly target protecting the data security and preserving the privacy of vehicle identities and trajectories in IoV data aggregation, while

keeping both the communication and computation costs of vehicles low. Our works not only address the above-proposed research issues but also open up some new research directions and possibilities for many applications. Details are provided in the corresponding chapters. We summarize our works, the remaining problems, and the possible future directions in Chapter 6.

Chapter 2

Lightweight Privacy-preserving Truth Discovery for Vehicular Air Quality Monitoring

Crowdsensing is a solution to collect Internet of Things (IoT) data from a large group of individual users (also called *participants* or *sources*). In this chapter, we introduce our work tackling the challenge of truth discovery in vehicular crowdsensing mentioned in Section 1.3. We first present an optimization-based truth discovery algorithm, which incorporates spatial and temporal correlations to solve the sparsity problem. A truth discovery framework is proposed, incorporating the data masking technique, anonymous communication and data perturbation. The framework is more lightweight than the existing cryptography-based methods. We also evaluate the work with simulations and fully discuss the performance and possible extensions.

The frequently used notations in this chapter are summarized in Table 2.1 for reference.

2.1 Introduction

Air pollution is a major health and environmental concern these years. However, performing fine-grained tasks is a challenge with the air quality monitoring stations deployed in practice. For example, citizens would like to know the latest best route with fresh air for cycling but the stations in use are usually inadequate. Vehicular crowdsensing (VCS) is one possible solution to accomplish such tasks. In VCS, ve-

Table 2.1: Notations in Chapter 2

Notation	Description
$G = \{g_1, g_2, \ldots, g_m\}$	m disjoint grids
$S = \{s_1, s_2, \ldots, s_n\}$	n sources
$V_s = \{v_{s,1}, v_{s,2}, \ldots, v_{s,c}\}$	c observation values generated by source s
$g^{s,j}$	The grid where $v_{s,j}$ is generated
t	A specific sensing cycle
$w_{s,t}$	The weight of s at sensing cycle t
$w'_{s,t}$	A temporary value of $w_{s,t}$
$W_s = \{w_{s,1}, w_{s,2}, \ldots, w_{s,t-1}\}$	The historical weights of s at t
$v^*_{g,t}$	The estimated ground truth of g at sensing cycle t
$v^{*\prime}_{g,t}$	A temporary value of $v^*_{g,t}$
$T_g = \{v^*_{g,1}, v^*_{g,2}, \ldots, v^*_{g,t-1}\}$	The historical truths of g at t
$\theta_{s,j,g}$	The spatial correlation between $g^{s,j}$ and g
$\mathrm{Dis}(g^{s,j}, g)$	The logical distance between $g^{s,j}$ and g
$\mathrm{D}_1(g^{s,j}, g)$	The geographical distance between $g^{s,j}$ and g
$\mathrm{D}_2(w_{s,t}, w_{s,i})$	The temporal distance between $w_{s,t}$ and $w_{s,i}$
$\mathrm{D}_3(v_{s,j}, v^*_{g,t})$	The deviation between $v_{s,j}$ and $v^*_{g,t}$
α	Masks
β	The masked values
χ	The summations of specific values
p_1, p_2	The probabilities used in the grid perturbation
ψ_1	The Laplace noise used in the grid perturbation
ψ_2	The Laplace noise used in the value perturbation
$\hat{v_{s,j}}$	The perturbed value of $v_{s,j}$
PID_s	The pseudo-ID of s
RID_s	The real ID of s

hicles equipped with onboard sensors collect the data from each block or street of a city. The data are uploaded to a server at periodic intervals. As a result, the server can update the air quality values at a fine granularity.

In VCS, one typical challenge is truth discovery. To be specific, the sensory data provided by vehicles usually vary in quality because of the different precisions of onboard sensors and possibly malicious behaviors of the drivers. Thus, discovering the reliability of participants which is unknown a priori from the biased or fake data

is of significant importance. The process of finding the true results of the task and the reliability of each participant is called truth discovery.

Many studies focusing on truth discovery have been conducted in recent years [52, 118, 28]. The approaches they propose usually need a large amount of data to gain high accuracy. However, in real life, only a small portion of blocks have very high traffic while a large number of blocks cannot provide adequate data [117], which is referred to as the *long tail phenomenon*. This data sparsity problem may result in inaccurate reliability discovery and truth finding. Additionally, the trajectories of vehicles, containing sensitive information such as the locations of home and companies, may be revealed in the process. Protecting the privacy is a challenge in the truth discovery of vehicular crowdsensing.

We first propose a truth discovery algorithm, **ST** (**S**patial and **T**emporal) to handle the data sparsity problem. The intuitions are threefold: a) neighbor blocks or streets are likely to have similar air quality owing to the dispersion of atmospheric pollutants; b) the current quality value can be predicted from the historical data because the change of air quality usually takes time; c) the historical reliability of participants can be utilized to help estimate the current reliability. To further improve the performance, the ST algorithm used is simplified for truth discovery with sufficient data. To protect both the observation values and trajectories of vehicles, we propose a privacy-preserving framework, named **EAirQ**. Note that we proposed the previous version of EAirQ in 2019 [63], which is named **AirQ** (**Air Q**uality monitoring), so that we call the framework in this chapter EAirQ (**E**nhanced **AirQ**). In EAirQ, the essential data are masked and perturbed before uploading. An anonymous authentication scheme is adopted. Simulation results show that EAirQ works well in fine-grained air quality monitoring while also maintaining the privacy-preserving property.

The main work and contributions are as follows:

- We present an optimization-based truth discovery algorithm, ST. Spatial and temporal correlations are combined to solve the data sparsity problem, which makes the algorithm suitable for fine-grained tasks.
- Two different truth discovery algorithms are combined to reduce the negative impact of data reuse when there are sufficient data.
- We present a privacy-preserving framework, EAirQ, based on the technique of masking, anonymous authentication and perturbation. We protect the observation values and the trajectories of vehicles in the process of truth discovery.

- Different from the existing privacy-preserving methods based on cryptography, EAirQ is lightweight from the perspective of computation and communication costs on vehicles. Thus, it is suitable for vehicular networks.

The rest of the chapter is organized as follows: in Sections 2.2 and 2.3, we show the related work and introduce the cryptography techniques. The problem is formally defined in Section 2.4 The details of the proposed ST algorithm and EAirQ framework are provided in Section 2.5. We conduct a series of experiments and analyze the privacy of the proposed framework in Section 2.6. In Section 2.7, the possible scenarios, the extensions and the remaining issues are discussed. We conclude the work in Section 2.8.

2.2 Related Work

Crowdsensing and truth discovery have become hot topics these years [81]. To solve the sparsity problem, Zhang *et al.* [125] present a robust truth discovery scheme which quantifies the attitude that human expressed and incorporates the historical contributions. Yang *et al.* [120] incorporate the information about the social network in a truth discovery framework and develop Laplace variational inference methods to estimate participants' reliabilities. Purahoo *et al.* [86] design a crowdsensing app for air and noise pollution detection, fire detection and smart parking, where data are collected from the built-in smartphone sensors and the IoT sensors in fixed locations of a city. Ren [93] discuss how to improve the robustness and efficiency of vehicular crowdsensing and give an example of participant selection. However, these schemes either do not fully consider the challenges in vehicular networks or cannot well adapt to fine-grained air quality monitoring.

Multi-party secure computation techniques have been adopted to address the data privacy issues [73, 119, 112, 54, 56]. Some techniques, such as Yao's garbled circuits, are not suitable for the data aggregation scenario with a large number of vehicles. A typical choice in VANETs and IoT is homomorphic encryption. Kong *et al.* [54] adopt the modified Paillier cryptosystem for privacy-preserving sensory data collection in VANETs. Li *et al.* [56] improve the labeled homomorphic encryption (LabHE) cryptosystem to tackle the privacy challenge among the data owners, untrustworthy servers, and the data users in Industrial Internet of Things (IIoT). Miao *et al.* [73] propose a mechanism called PPTD based on the threshold Paillier cryptosystem as well.

The proposed scheme can preserve the privacy of weights and observation values in truth discovery. However, considerable amounts of cryptography-based calculations have to be conducted by participants in these works, which is a common limitation of cryptosystem-based schemes.

2.3 Cryptography Tools

Data masking: data masking allows a server to aggregate data from client parties in a secure way. In an additive masking algorithm, all sensitive inputs are masked by adding random values called *masks*. The randomness should be canceled once the masked inputs are aggregated. Thus, the server only learns the sum of the clients' inputs.

In this work, we adopt a masking algorithm with one-time pads [15]. In this algorithm, suppose the set of client parties is U. An input from client $a \in U$ is denoted as x_a. Each pair of clients (a, b) that satisfies $a < b$ agrees on a mask $\alpha_{a,b}$. Note that we use $a < b$ to represent that the index of client a is smaller than that of b for simplicity. Then the masked value of x_a can be represented as:

$$y_a = x_a + \sum_{b \in U : a < b} \alpha_{a,b} - \sum_{b \in U : a > b} \alpha_{b,a}. \tag{2.1}$$

After collecting all the masked values, the server can compute the sum of $\{x_a \mid a \in U\}$ as follows:

$$\begin{aligned} \sum_{a \in U} y_a &= \sum_{a \in U} \left(x_a + \sum_{b \in U : a < b} \alpha_{a,b} - \sum_{b \in U : a > b} \alpha_{b,a} \right) \\ &= \sum_{a \in U} x_a. \end{aligned} \tag{2.2}$$

In the algorithm of [15], each pair of clients (a, b) should exchange secrets to reach an agreement on the mask $\alpha_{a,b}$, which brings high communication cost. Besides, dropping clients after masking will result in failure of the summation. However, these are not problems in our work, which will be described in details in Section 2.5.3.

Anonymous authentication: a digital signature-based anonymous authentication scheme provides not only the integrity and authenticity of a message, but also

the anonymity of the signer. It can be achieved by using pseudo-IDs. To be specific, a message sender (or signer) uses a pseudo-ID in the signing process, issued by a trusted third party, as a replacement of the real ID. The message receiver can verify the message sent but cannot trace the real identity of the sender.

Moni *et al.* [77] present a distributed, scalable and low-overhead authentication scheme for vehicular ad-hoc networks (VANETs). The scheme has two layers: the upper layer of the trusted authority (TA) and regional trusted authorities (RTAs), and the lower layer of vehicles and road-side units (RSUs). In this work, vehicles are issued with pseudo-IDs by RTAs. Messages are then signed with the pseudo-IDs by the RSA algorithm. Only TA and RTAs have the ability to reveal the real identities of vehicles. In our work, the scheme is adopted to achieve the anonymous communication between a vehicle and an RSU. We omit the description of the parameter distribution, the communication establishment and the signature construction in this chapter for brevity. For more details, we refer readers to [77].

Randomized response and local differential privacy: randomized response is a survey technique that allows surveyees to respond to sensitive questions while maintaining the confidentiality. A randomization device (e.g., a coin flip) is used by surveyees to decide if an answer should be given truthfully. The interviewer can get a reliable statistic result from the biased answers.

Local differential privacy (LDP) is a model to protect individuals' privacy in statistical computations. Different from differential privacy, there is no trusted central server (i.e., a data collector) in LDP because participants perturb the raw data locally.

Although we do not adopt any specific randomized response or LDP algorithms in our work, the ideas extracted from the two techniques are used to develop a perturbation mechanism. Details can be found in Section 2.5.4.

2.4 Problem Definition

In crowdsensing systems, there are usually two types of parties: *sources* and a *server*. Sources are the participants who conduct sensing tasks and then upload the sensory

data to a server for further processing and analysis. The sensory readings for a specific sensing task are called *observation values*. The actual true value of a task is denoted by *ground truth*. In truth discovery algorithms, *weight* represents the reliability of a source. *Truth* denotes the estimated ground truth of a task based on the collected sensory data and weights.

We formally define the problem targeted as follows:

We divide the urban area to m disjoint *grids* $G = \{g_1, g_2, \ldots, g_m\}$, typically streets or blocks in practice. A *sensing cycle* is a static time slot (e.g., 15 minutes). In each sensing cycle, the sensory data are uploaded once. Then the truths and weights can be updated based on the data. The concrete crowdsensing task is to get an estimated air quality value for each grid in each sensing cycle.

There are n sources, i.e., vehicles, registered in the system denoted by $S = \{s_1, s_2, \ldots, s_n\}$. Note that we use the terms "source" and "vehicle" interchangeably in the following. A source s provides a *report*, containing a certain number (denoted by c) of observation values, $V_s = \{v_{s,1}, v_{s,2}, \ldots, v_{s,c}\}$, in each sensing cycle. The j-th observation value is denoted as $v_{s,j}$ where $j \in \{1, 2, \ldots, c\}$. The grid where $v_{s,j}$ is generated is denoted as $g^{s,j}$. We assume that each source provides at most one observation value for each grid. The weight of s at sensing cycle t is denoted as $w_{s,t}$, which combines the temporary weight $w'_{s,t}$ and the historical weights $W_s = \{w_{s,1}, w_{s,2}, \ldots, w_{s,t-1}\}$. Similarly, the estimated ground truth of g at sensing cycle t is denoted as $v^*_{g,t}$, which combines the temporary truth $v^{*\prime}_{g,t}$ and the historical truths $T_g = \{v^*_{g,1}, v^*_{g,2}, \ldots, v^*_{g,t-1}\}$. Note that for simplicity, we omit some subscripts. For example, we use s to denote s_i where $i \in \{1, 2, \ldots, n\}$. Our goal is to let the server calculate $v^*_{g,t}$ for each g and $w_{s,t}$ for each s in each sensing cycle t.

A trusted manager is introduced in our framework. It manages all vehicles and is considered to be honest and trustworthy. More introduction is given in Section 2.5.5. We assume that all the other parties except the trusted manager are semi-honest. To be specific, all the parties except the trusted manager follow the protocol of the proposed frameworks but may try to infer the sensitive information of other parties from the reports. The privacy-preserving goal in this work is that any observation value $v_{s,j}$ and the trajectory of vehicle s should not be revealed from the reports to any semi-honest party except s itself.

In addition, we assume the communications among all parties are reliable. All packets can be sent and received successfully in the network. The communication performance and the quality of service (QoS), such as the packet loss rate, are not

considered. Thus, we can focus on the truth discovery performance and the security and privacy-preservation goals in the application layer.

2.5 Framework Design

2.5.1 ST algorithm

To address the data sparsity problem, we take the spatial correlation of grids and the temporal correlations of weights and truths into consideration. We first introduce the details of the correlations and then propose the optimization problem.

Spatial Correlation

When calculating the estimated ground truth $v^*_{g,t}$, not only the observation values provided for g but also the values for other grids are used. The correlation between two grids is represented by a parameter $\theta_{s,j,g}$, i.e., how much we can rely on $v_{s,j}$ for $v^*_{g,t}$. The intuition is that the nearer the two grids are, the more likely they have similar air quality. Thus, $\theta_{s,j,g}$ is calculated by the *logical distance* $\mathrm{Dis}(g^{s,j}, g)$ of the two grids $g^{s,j}$ and g. We adopt the *Gaussian kernel* for $\mathrm{Dis}(g^{s,j}, g)$ as follows [72, 59]:

$$\mathrm{Dis}(g^{s,j}, g) = \begin{cases} \exp(-\frac{\mathrm{D}_1(g^{s,j},g)^2}{2\omega^2}), & \text{if } \mathrm{D}_1(g^{s,j}, g) < u \\ 0, & \text{otherwise} \end{cases} \tag{2.3}$$

where ω is the width parameter of the kernel and u is a threshold we set. $\mathrm{D}_1(g^{s,j}, g)$ is the *geographical distance* between $g^{s,j}$ and g, which is the orthodromic distance in our work. Orthodromic distance is the shortest distance between two points on the surface of the earth and can be calculated with the longitudes and latitudes. The logical distance decreases with the geographical distance and ranges between 0 and 1, which makes it easy to handle the spatial correlation.

Note that, to take full advantage of the correlations between grids, $\theta_{s,j,g}$ can be calculated by not only the logical distance $\mathrm{Dis}(g^{s,j}, g)$ but also the *similarity distance*. The extensibility of the work will be discussed in Section 2.7.

Temporal Correlation

A source who performed bad, i.e., had low weights, in the past, is likely to be unreliable in the current sensing cycle. In other words, the historical weights of a source can be

used to predict the latest weight. Based on this intuition, we define $w_{s,t}$ as:

$$\begin{aligned} w_{s,t} &= \mathrm{F}_1(w'_{s,t}, W_s) \\ &= \begin{cases} \frac{\sum_{i=1}^{t-1} k_i w_{s,i} + k_t w'_{s,t}}{\sum_{i=1}^{t} k_i}, & \text{if } W_s \neq \emptyset \\ w'_{s,t}, & \text{otherwise} \end{cases} \end{aligned} \tag{2.4}$$

where the function F_1 combines the historical weights W_s and the temporary weight $w'_{s,t}$ of source s with the *Inverse distance weighting* method. k_i is defined as:

$$k_i = \frac{1}{\mathrm{D}_2(w_{s,t}, w_{s,i})^{\rho_w}}. \tag{2.5}$$

ρ_w is a positive real number, called the power parameter. It controls the degree of dependence on historical weights. $\mathrm{D}_2(w_{s,t}, w_{s,i})$ is the *temporal distance* between two data points, i.e., $w_{s,t}$ and $w_{s,i}$. In other words, it represents the time interval between the past and current sensing cycles. $\mathrm{D}_2(w_{s,t}, w_{s,i})$ can be calculated as:

$$\mathrm{D}_2(w_{s,t}, w_{s,i}) = t - i + 1. \tag{2.6}$$

The parameter k_i results in a negative correlation between the temporal distances and the significance of past weights in weight updating.

Similarly, the ground truths of grids are not only related to the observation values but also the past records because the easing of air pollution takes time. Based on this intuition, we define $v^*_{g,t}$ as:

$$\begin{aligned} v^*_{g,t} &= \mathrm{F}_2(v^{*\prime}_{g,t}, T_g) \\ &= \begin{cases} \frac{\sum_{i=1}^{t-1} k'_i v^*_{g,i} + k'_t v^{*\prime}_{g,t}}{\sum_{i=1}^{t} k_i}, & \text{if } T_g \neq \emptyset \\ v^{*\prime}_{g,t}, & \text{otherwise} \end{cases} \end{aligned} \tag{2.7}$$

where

$$k'_i = \frac{1}{\mathrm{D}_2(v_{g,t}, v_{g,i})^{\rho_t}}. \tag{2.8}$$

The power parameter ρ_t controls the degree of dependence on historical truths. With k'_i, the newer historical truths are more significant to the estimation of the current truth for each grid. For simplicity, we transform (2.4) and (2.7) to:

$$v^*_{g,t} = \delta_{1,g} + \delta_{2,g} v^{*\prime}_{g,t}. \tag{2.9}$$

$$w_{s,t} = \delta_{3,s} + \delta_{4,s} w'_{s,t}. \tag{2.10}$$

Optimization Problem

Algorithm 2.1. Truth discovery algorithm: ST

Input: Observation values from n sources: $\{V_s \mid s \in S\}$, historical truths of m grids: $\{T_g \mid g \in G\}$, historical weights from n sources: $\{W_s \mid s \in S\}$, and parameters: $\{\theta_{s,j,g} \mid s \in S, j \in \{1, 2, \ldots, c\}, g \in G\}$

Output: Estimated ground truths for m grids: $\{v^*_{g,t}\}$, weights for n sources: $\{w_{s,t}\}$, and updated records: $\{T_g \mid g \in G\}$ and $\{W_s \mid s \in S\}$

1: Initialize $v^{*\prime}_{g,t}$ for each grid g to the average of all the observation values provided for the grid;
2: Initialize $w'_{s,t}$ for each source s to $\frac{1}{n}$;
3: Calculate $\delta_{1,g}$ and $\delta_{2,g}$ based on W_s for each source s (i.e., (2.4) and (2.9));
4: Calculate $\delta_{3,s}$ and $\delta_{4,s}$ based on T_g for each grid g (i.e., (2.7) and (2.10));
5: **repeat**
6: **for** each source s **do**
7: Update $w'_{s,t}$ based on $\delta_{3,s}$, $\delta_{4,s}$, $\{\theta_{s,j,g} \mid s \in S, j \in \{1, 2, \ldots, c\}, g \in G\}$, $\{V_s \mid s \in S\}$ and $\{v^{*\prime}_{g,t}\}$ (i.e., (2.16));
8: **end for**
9: **for** each grid g **do**
10: Update $v^{*\prime}_{g,t}$ based on $\delta_{1,g}$, $\delta_{2,g}$, $\{\theta_{s,j,g} \mid s \in S, j \in \{1, 2, \ldots, c\}\}$, $\{V_s \mid s \in S\}$ and $\{w'_{s,t}\}$ (i.e., (2.15));
11: **end for**
12: **until** the convergence criterion is satisfied;
13: Update $v^*_{g,t}$ for each g based on $v^{*\prime}_{g,t}$, $\delta_{1,g}$ and $\delta_{2,g}$ (i.e., (2.9));
14: Update $w_{s,t}$ for each s based on $w'_{s,t}$, $\delta_{3,s}$ and $\delta_{4,s}$ (i.e., (2.10));
15: Append $v^*_{g,t}$ to T_g for each g;
16: Append $w_{s,t}$ to W_s for each s;
17: **return** $\{v^*_{g,t}\}$, $\{w_{s,t}\}$, $\{T_g \mid g \in G\}$ and $\{W_s \mid s \in S\}$

Taking full advantage of the above correlations, we define a cost function as follows:

$$\mathrm{J} = \sum_{s \in S} \sum_{g \in G} \sum_{j \in \{1,2,\ldots,c\}} \mathrm{F}_1(w'_{s,t}, W_s) \theta_{s,j,g} \, \mathrm{D}_3(v_{s,j}, \mathrm{F}_2(v^{*\prime}_{g,t}, T_g)) \tag{2.11}$$

where $\theta_{s,j,g}$ controls the reuse of sensory data. Functions F_1 and F_2 combine the historical records with the temporary results. $\mathrm{D}_3(v_{s,j}, \mathrm{F}_2(v^{*\prime}_{g,t}, T_g))$ is the deviation between an observation value $v_{s,j}$ and the estimated ground truth $\mathrm{F}_2(v^{*\prime}_{g,t}, T_g)$ (or $v^*_{g,t}$). We use *truth distance* to represent the deviation. The squared L2-norm is adopted to

calculate it as:

$$\mathrm{D}_3(v_{s,j}, \mathrm{F}_2(v^{*\prime}_{g,t}, T_g)) = \|v_{s,j} - \mathrm{F}_2(v^{*\prime}_{g,t}, T_g)\|^2. \tag{2.12}$$

To guarantee the convexity of the optimization problem, we adopt a constraint as:

$$\sum_{s\in S} \exp(-\mathrm{F}_1(w'_{s,t}, W_s)) = 1 \tag{2.13}$$

which regularizes the value of $w_{s,t}$ by constraining the sum of $\exp(-w_{s,t})$ [55].

Therefore, the optimization problem for truth discovery can be formulated as follows:

$$\begin{aligned} &\min_{\{w'_{s,t}\},\{v^{*\prime}_{g,t}\}} \sum_{s\in S}\sum_{g\in G}\sum_{j\in\{1,2,\ldots,c\}} \mathrm{F}_1(w'_{s,t}, W_s)\theta_{s,j,g}\,\mathrm{D}_3(v_{s,j}, \mathrm{F}_2(v^{*\prime}_{g,t}, T_g)), \\ &\text{s.t.} \sum_{s\in S} \exp(-\mathrm{F}_1(w'_{s,t}, W_s)) = 1. \end{aligned} \tag{2.14}$$

Solving the above convex optimization problem by KKT (Karush–Kuhn–Tucker) conditions, we have:

$$v^{*\prime}_{g,t} = \frac{\sum_{s\in S}\sum_{j\in\{1,2,\ldots,c\}} \mathrm{F}_1(w'_{s,t}, W_s)\theta_{s,j,g}(v_{s,j} - \delta_{1,g})}{\delta_{2,g}\sum_{s\in S}\sum_{j\in\{1,2,\ldots,c\}} \mathrm{F}_1(w'_{s,t}, W_s)\theta_{s,j,g}}. \tag{2.15}$$

$$w'_{s,t} = \frac{1}{\delta_{4,s}}\Big(\log\left(\frac{\sum_{s\in S}\sum_{g\in G}\sum_{j\in\{1,2,\ldots,c\}} \theta_{s,j,g}\|v_{s,j} - \mathrm{F}_2(v^{*\prime}_{g,t}, T_g)\|^2}{\sum_{g\in G}\sum_{j\in\{1,2,\ldots,c\}} \theta_{s,j,g}\|v_{s,j} - \mathrm{F}_2(v^{*\prime}_{g,t}, T_g)\|^2}\right) - \delta_{3,s}\Big). \tag{2.16}$$

Based on the solution above, we summarize the ST algorithm in Alg. 2.1. In each sensing cycle t, we update the temporary weights $\{w'_{s,t}\}$ and estimated ground truths $\{v^{*\prime}_{g,t}\}$ by (2.15) and (2.16) iteratively until the convergence criterion is satisfied (i.e., Steps 5–10). Final values $\{w_{s,t}\}$ and $\{v^{*}_{g,t}\}$ are calculated by (2.10) and (2.9) and appended to W_s and T_g, respectively (i.e., Steps 11–14).

2.5.2 SST algorithm

Note that, in our previous work, we observe that the ST algorithm works good with insufficient data but the data reuse may bring a negative impact to the truth discovery performance with sufficient data [63]. Thus, we simplify the ST truth discovery algorithm to **SST** (**S**implified **ST**) as a substitution of ST when there are sufficient

reports. The corresponding optimization problem is defined as:

$$\begin{aligned} &\min_{\{w'_{s,t}\},\{v^*_{g,t}\}} \sum_{s\in S}\sum_{g\in G}\sum_{j\in\{1,2,\ldots,c\}\wedge g^{s,j}=g} \mathrm{F}_1(w'_{s,t}, W_s)\,\mathrm{D}_3(v_{s,j}, v^*_{g,t}), \\ &\text{s.t.} \sum_{s\in S}\exp(-\mathrm{F}_1(w'_{s,t}, W_s)) = 1. \end{aligned} \tag{2.17}$$

In this problem, we only involve the reliability of a report (i.e., the weight of the data provider w_s) and the temporal correlation of weights (i.e., the historical weights W_s). Reports are not reused among grids. The historical truths are not considered as well. The intuition is in two aspects: a) when there are sufficient reports, the average value can reflect the ground truth accurately; and b) ST may cause a deviation from the average because of the incorporation of the correlations.

Solving the above convex optimization problem by KKT conditions, we have:

$$v^*_{g,t} = \frac{\sum_{s\in S}\sum_{j\in\{1,2,\ldots,c\}\wedge g^{s,j}=g} \mathrm{F}_1(w'_{s,t}, W_s) v_{s,j}}{\sum_{s\in S}\mathrm{F}_1(w'_{s,t}, W_s)}. \tag{2.18}$$

$$w'_{s,t} = \frac{1}{\delta_{4,s}}\Big(\log\left(\frac{\sum_{s\in S}\sum_{g\in G}\sum_{j\in\{1,2,\ldots,c\}\wedge g^{s,j}=g}\|v_{s,j}-v^*_{g,t}\|^2}{\sum_{g\in G}\sum_{j\in\{1,2,\ldots,c\}\wedge g^{s,j}=g}\|v_{s,j}-v^*_{g,t}\|^2}\right) - \delta_{3,s}\Big). \tag{2.19}$$

The SST algorithm is described in Alg. 2.2. In each sensing cycle t, $\{w'_{s,t}\}$ and $\{v^*_{g,t}\}$ are updated by (2.18) and (2.19) iteratively until the convergence criterion is satisfied (i.e., Steps 5–9). Final values $\{w_{s,t}\}$ and $\{v^*_{g,t}\}$ are appended to W_s and T_g, respectively (i.e., Steps 10–12). Note that although the historical truths $\{T_g \mid g \in G\}$ are not used in SST, they are necessary to be recorded for the EAirQ framework. More details are given in Section 2.5.5.

It is a challenge to preserve the privacy in both ST and SST while keeping the framework as lightweight as possible. To overcome the challenge, we adopt the masking technique in ST, which will be introduced in Section 2.5.3 and present a new perturbation mechanism for SST, which will be introduced in Section 2.5.4.

2.5.3 Masking mechanism

Before uploading reports in every sensing cycle, each vehicle s masks three types of values $\{\theta_{s,j,g}\cdot v_{s,j} \mid j\in\{1,2,\ldots,c\}\}$, $\{\theta_{s,j,g}\cdot v^2_{s,j} \mid j\in\{1,2,\ldots,c\}\}$, and $\{\theta_{s,j,g} \mid j\in\{1,2,\ldots,c\}\}$ for each grid g by the masking algorithm with one-time pads [15].

Algorithm 2.2. Truth discovery algorithm: SST

Input: Pairs of observation values and corresponding grids from n sources: $\{(v_{s,j}, g^{s,j}) \mid s \in S, j \in \{1, 2, \ldots, c\}\}$, and historical weights of n sources: $\{W_s \mid s \in S\}$
Output: Estimated ground truths for m grids: $\{v^*_{g,t}\}$, weights for n sources: $\{w_{s,t}\}$, and updated records: $\{T_g \mid g \in G\}$ and $\{W_s \mid s \in S\}$

1: Initialize $v^*_{g,t}$ for each grid g to the average of all the observation values provided for the grid;
2: Initialize $w'_{s,t}$ for each source s to $\frac{1}{n}$;
3: Calculate $\delta_{3,s}$ and $\delta_{4,s}$ based on T_g for each grid g (i.e., (2.7) and (2.10));
4: **repeat**
5: **for** each source s **do**
6: Update $w'_{s,t}$ based on $\delta_{3,s}$, $\delta_{4,s}$, and $\{(v_{s,j}, g^{s,j}) \mid j \in \{1, 2, \ldots, c\}\}$ (i.e., (2.19));
7: **end for**
8: **for** each grid g **do**
9: Update $v^*_{g,t}$ based on $\{(v_{s,j}, g^{s,j}) \mid s \in S, j \in \{1, 2, \ldots, c\} \wedge g^{s,j} = g\}$ and $\{w'_{s,t}\}$ (i.e., (2.18));
10: **end for**
11: **until** the convergence criterion is satisfied;
12: Update $w_{s,t}$ for each s based on $w'_{s,t}$, $\delta_{3,s}$ and $\delta_{4,s}$ (i.e., (2.10));
13: Append $v^*_{g,t}$ to T_g for each g;
14: Append $w_{s,t}$ to W_s for each s;
15: **return** $\{v^*_{g,t}\}$, $\{w_{s,t}\}$, $\{T_g \mid g \in G\}$ and $\{W_s \mid s \in S\}$

In this work, a significant difference is that each vehicle s chooses masks for pairs of specific values it maintains. Thus, there is no need for secret exchange protocols when masking and dropping clients will not impede the truth discovery process. The difference guarantees the low computation and communication costs on the vehicle-side and the availability of the framework.

We use $\{\theta_{s,j,g} \cdot v_{s,j} \mid j \in \{1, 2, \ldots, c\}\}$ as an example to describe the masking process in details. For each pair of $(\theta_{s,j,g} \cdot v_{s,j}, \theta_{s,j',g} \cdot v_{s,j'})$ that satisfies $j < j'$, s generates a random value $\alpha^{s,g}_{j,j'}$ by a pseudo-random number generator (PRNG). We use $\beta^{s,j,g}_1$ to denote the masked $\theta_{s,j,g} \cdot v_{s,j}$. It can be calculated based on (2.1) as follows:

$$\beta^{s,j,g}_1 = \theta_{s,j,g} \cdot v_{s,j} + \sum_{j' \in \{1,2,\ldots,c\}: j < j'} \alpha^{s,g}_{j,j'} - \sum_{j' \in \{1,2,\ldots,c\}: j > j'} \alpha^{s,g}_{j',j}. \tag{2.20}$$

Based on (2.2), $\chi^{s,g}_1$, the sum of $\{\theta_{s,j,g} \cdot v_{s,j} \mid j \in \{1, 2, \ldots, c\}\}$ can be calculated

as:

$$\begin{aligned}\chi_1^{s,g} &= \sum_{j\in\{1,2,\ldots,c\}} \theta_{s,j,g}\cdot v_{s,j} \\ &= \sum_{j\in\{1,2,\ldots,c\}} \beta_1^{s,j,g}.\end{aligned} \tag{2.21}$$

Similarly, we denote the masked $\theta_{s,j,g}\cdot v_{s,j}^2$ as $\beta_2^{s,j,g}$ and the masked $\theta_{s,j,g}$ as $\beta_3^{s,j,g}$. Then, the following equations are satisfied:

$$\begin{aligned}\chi_2^{s,g} &= \sum_{j\in\{1,2,\ldots,c\}} \theta_{s,j,g}\cdot v_{s,j}^2 \\ &= \sum_{j\in\{1,2,\ldots,c\}} \beta_2^{s,j,g}.\end{aligned} \tag{2.22}$$

$$\begin{aligned}\chi_3^{s,g} &= \sum_{j\in\{1,2,\ldots,c\}} \theta_{s,j,g} \\ &= \sum_{j\in\{1,2,\ldots,c\}} \beta_3^{s,j,g}.\end{aligned} \tag{2.23}$$

Based on (2.21), (2.22) and (2.23), (2.15) and (2.16) can be transformed as follows:

$$\begin{aligned}v_{g,t}^{*\prime} &= \frac{\sum_{s\in S}\sum_{j\in\{1,2,\ldots,c\}} \mathrm{F}_1(w_{s,t}', W_s)\theta_{s,j,g}(v_{s,j}-\delta_{1,g})}{\delta_{2,g}\sum_{s\in S}\sum_{j\in\{1,2,\ldots,c\}} \mathrm{F}_1(w_{s,t}', W_s)\theta_{s,j,g}} \\ &= \frac{\sum_{s\in S}\mathrm{F}_1(w_{s,t}', W_s)\Big(\sum_{j\in\{1,2,\ldots,c\}}\theta_{s,j,g}v_{s,j} - \delta_{1,g}\sum_{j\in\{1,2,\ldots,c\}}\theta_{s,j,g}\Big)}{\delta_{2,g}\sum_{s\in S}\mathrm{F}_1(w_{s,t}', W_s)\sum_{j\in\{1,2,\ldots,c\}}\theta_{s,j,g}} \\ &= \frac{\sum_{s\in S}\mathrm{F}_1(w_{s,t}', W_s)\Big(\chi_1^{s,g} - \delta_{1,g}\chi_3^{s,g}\Big)}{\delta_{2,g}\sum_{s\in S}\mathrm{F}_1(w_{s,t}', W_s)\chi_3^{s,g}}.\end{aligned} \tag{2.24}$$

$$\begin{aligned}w_{s,t}' =& \frac{1}{\delta_{4,s}}\Big(\log\left(\frac{\sum_{s\in S}\sum_{g\in G}\sum_{j\in\{1,2,\ldots,c\}}\theta_{s,j,g}\|v_{s,j}-\mathrm{F}_2(v_{g,t}^{*\prime},T_g)\|^2}{\sum_{g\in G}\sum_{j\in\{1,2,\ldots,c\}}\theta_{s,j,g}\|v_{s,j}-\mathrm{F}_2(v_{g,t}^{*\prime},T_g)\|^2}\right) - \delta_{3,s}\Big) \\ =& \frac{1}{\delta_{4,s}}\Big(\log\left(\frac{\sum_{s\in S}\sum_{g\in G}\chi_2^{s,g} - 2\,\mathrm{F}_2(v_{g,t}^{*\prime},T_g)\cdot\chi_1^{s,g} + \mathrm{F}_2(v_{g,t}^{*\prime},T_g)^2\chi_3^{s,g}}{\sum_{g\in G}\chi_2^{s,g} - 2\,\mathrm{F}_2(v_{g,t}^{*\prime},T_g)\cdot\chi_1^{s,g} + \mathrm{F}_2(v_{g,t}^{*\prime},T_g)^2\chi_3^{s,g}}\right) - \delta_{3,s}\Big).\end{aligned} \tag{2.25}$$

Therefore, the server can update $\{v_{g,t}^*\}$ and $\{w_{s,t}\}$ only with the collected $\{\beta_1^{s,j,g} \mid j\in\{1,2,\ldots,c\}, g\in G\}$, $\{\beta_2^{s,j,g} \mid j\in\{1,2,\ldots,c\}, g\in G\}$ and $\{\beta_3^{s,j,g} \mid j\in\{1,2,\ldots,c\}, g\in G\}$ from each vehicle s. In other words, both the observation values

$\{V_s\}$ and $\{\theta_{s,j,g}\}$ which maintain the information of the vehicle trajectories, are not revealed to the server.

2.5.4 Perturbation mechanism

Different from ST, the pairs of observation values and corresponding grids $\{(v_{s,j}, g^{s,j}) \mid j \in \{1, 2, \ldots, c\}\}$ are uploaded by each s in SST. It may reveal both the real observation values and the trajectories. The masking technique adopted for ST is not suitable for SST because the inputs of the algorithms are different. Thus, we presented the perturbation mechanism for SST, which is inspired by the idea of randomized response and LDP. The mechanism adds a two-layer perturbation to the raw data as follows:

Grid perturbation: similar to the idea of randomized response, vehicles do not always truthfully provide the trajectories. In other words, each vehicle perturbs the records of grids it passed by as follows: a) for each observation value $v_{s,j}$ provided by s in sensing cycle t, s removes it from the list V_s with a probability p_1 (e.g., 0.2), as defined in (2.26); b) for each grid g that satisfies $\{g^{s,j} \neq g \mid j \in \{1, \ldots, c\}\}$, s adds $v_{s,c+1}$ to V_s with the probability p_2. $v_{s,c+1}$ is calculated by (2.27).

$$v_{s,j} = \begin{cases} v_{s,j}, & \text{with probability } 1 - p_1 \\ \emptyset, & \text{with probability } p_1 \end{cases}. \tag{2.26}$$

$$v_{s,c+1} = v^*_{g,t-1} + \psi_1. \tag{2.27}$$

c is the current number of reports in V_s. ψ_1 is a Laplace noise generated from a Laplace distribution $\mathcal{L}(0, \lambda_1)$ where 0 is the location parameter and λ_1 is the scale parameter. Note that we suggest setting p_2 with a small value (such as 0.05) to reduce the impact on accuracy. In the following, we use the term, *imitated reports*, to denote the reports generated and added in the grid perturbation process.

Value perturbation: similar to the idea of LDP, each vehicle locally perturbs the observation values it provides. To be specific, for each $v_{s,j} \in V_s$, s adds a Laplace noise as follows:

$$\hat{v_{s,j}} = v_{s,j} + \psi_2 \tag{2.28}$$

where $\psi_2 \sim \mathcal{L}(0, \lambda_2)$ and λ_2 is the scale parameter.

In our work, the perturbation mechanism may involve bias to the truth discovery results. However, a moderate sacrifice of precision is acceptable when there are sufficient data. Besides, the privacy and precision can be balanced by adjusting the parameters based on different user demands and scenarios. More discussion and analysis are given in Section 2.6.

To provide further protection of the privacy, besides the masking and perturbation mechanisms, we adopt the anonymous communication between vehicles and RSUs. More details can be found in Section 2.5.5.

2.5.5 EAirQ framework

In EAirQ, there are four entities: vehicles (i.e., sources), RSUs, a server (i.e., the truth discovery server), and a trusted manager (TM). The TM not only acts as a TA introduced in Section 2.3, but also manages all vehicles. To be specific, it has but is not limited to the following functions: a) the TM generates and distributes the system parameters including the ones necessary for anonymous authentication; b) the TM can explore the real identity of a vehicle; c) the TM maintains the historical weights for all vehicles; d) the TM can update the weights for vehicles based on application-level user activities. For example, a user (i.e., a driver of a vehicle) who reads articles every day in the app for a long time is intuitively more reliable than a user who signed up several days before; and e) the TM works with cloud techniques and can communicate with the truth discovery server efficiently.

As shown in Fig. 2.1, in each sensing cycle, a vehicle s generates sensory data and collects necessary information from road-side units (RSUs) in the process of *data generation*. The sensory data is masked with the masking mechanism and perturbed with the perturbation mechanism separately in the processes of *data preprocessing*, The masked data and the perturbed data are expected to be sent to an RSU through anonymous communication in the *anonymous data uploading* process. After collecting all the data from RSUs, the server performs the truth discovery task with the help of the TM in the last process named *data handling*. Now we describe the above four processes in details as follows:

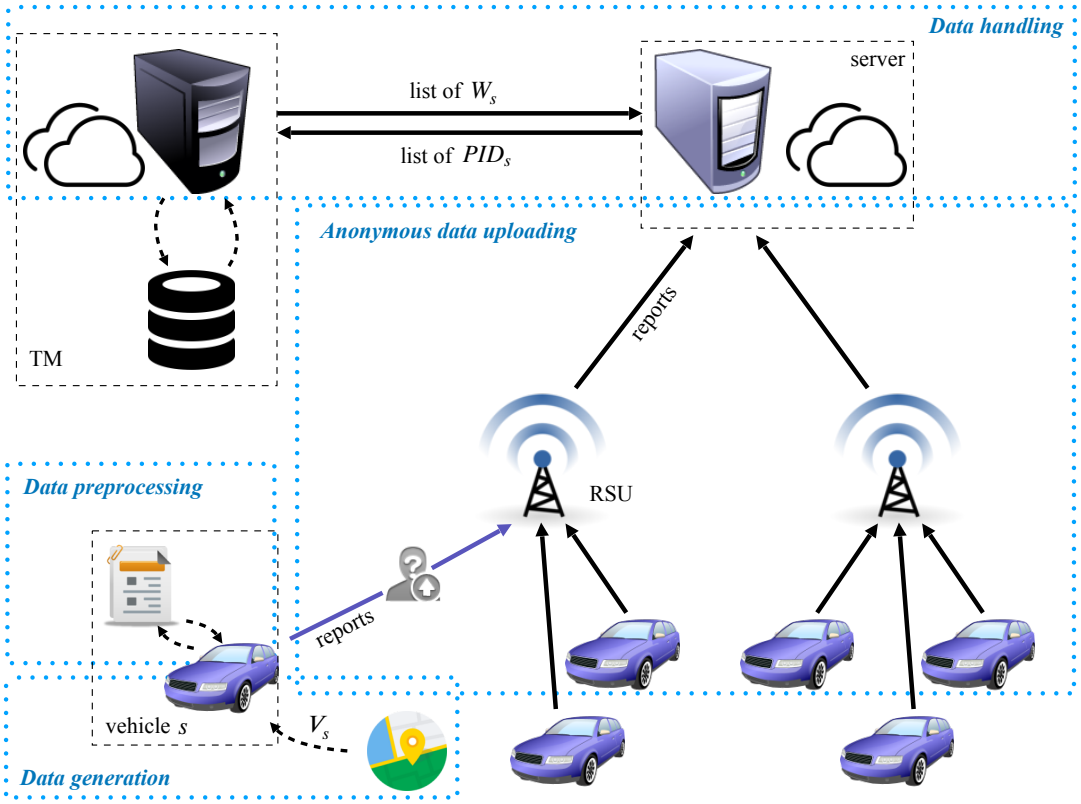


Figure 2.1: EAirQ framework processes

1. **Data generation**: in this process, each vehicle s generates observation values V_s for the grids passed by. Meanwhile, a vehicle s should ask for $\{\theta_{s,j,g} \mid g \in G\}$ from the nearest RSU for each $v_{s,j}$. Recall that $\theta_{s,j,g}$ represents the spatial correlation between $g^{s,j}$ and g, which is a constant. Thus, recording all the $\{\theta_{s,j,g} \mid g \in G\}$ by the nearest RSU of grid $g^{s,j}$ reduces the storage burden on vehicles. Besides, $\theta_{s,j,g}$ equals 0 when the two grids are far from each other and have insignificant spatial correlation. Therefore, RSUs only need to send the none-zero values to lower the communication cost.

   Besides, to get ready for the following processes, s should set up the anonymous communication with RSUs, i.e., update the pseudo-ID, PID_s, for message signing and verification [77]. Note that the processes of system parameter distribution and anonymous communication establishment are not shown in Fig. 2.1 as they are not the main focuses of this work.

2. **Data preprocessing**: in this process, s first masks $\{\theta_{s,j,g} \cdot v_{s,j} \mid j \in \{1, 2, \ldots, c\}\}$,

$\{\theta_{s,j,g} \cdot v_{s,j}^2 \mid j \in \{1, 2, \ldots, c\}\}$, and $\{\theta_{s,j,g} \mid j \in \{1, 2, \ldots, c\}\}$ for each grid g with the raw observation values, as described in Section 2.5.3. The masked values are denoted as $\beta_1^{s,j,g}$, $\beta_2^{s,j,g}$ and $\beta_3^{s,j,g}$.

In addition, s performs the grid perturbation and value perturbation on the raw observation values, as described in Section 2.5.4.

3. **Anonymous data uploading**: in each sensing cycle, a vehicle s only uploads data once to the server. The uploaded report contains $\{\beta_1^{s,j,g} \mid j \in \{1, 2, \ldots, c\}\}$, $\{\beta_2^{s,j,g} \mid j \in \{1, 2, \ldots, c\}\}$, $\{\beta_3^{s,j,g} \mid j \in \{1, 2, \ldots, c\}\}$ and $\{\hat{v_{s,j}} \mid g^{s,j} = g\}$ for each grid $g \in G$.

   To reduce the communication cost, s does not transfer the report to the server directly but to the nearest RSU at that time. The server then collects all the reports from all the RSUs. Recall that $\{\hat{v_{s,j}} \mid g^{s,j} = g\}$ is the perturbed data. The report is uploaded with the pseudo-ID PID_s. Thus, both the RSU and the server cannot link a report to a vehicle.

4. **Data handling**: the pseudo-IDs protect vehicles but bring a challenge to append the estimated weight $w_{s,t}$ to the corresponding vehicle's record W_s in each sensing cycle. Thus, after collecting all the reports from RSUs, the cloud server first requests W_s for each s from the TM by sending the list of PID_s. The TM looks for corresponding W_s by exploring the true identity (i.e., the real ID RID_s) of s with PID_s. One point worth mentioning is that, the TM only shares a list of W_s without RID_s to the server. Besides, the TM has the authority to update W_s based on the application-layer user activities so that W_s may change every sensing cycle. Thus, the server cannot obtain the corresponding real identities of the vehicles from the TM in the process.

   After acquiring W_s for all PID_s, the server estimates the weights and ground truths as follows: for grid g with sufficient reports, the ground truth $v_{g,t}^*$ is estimated by the simplified truth discovery algorithm SST with the perturbed data. For g with insufficient reports, $v_{g,t}^*$ is estimated by the truth discovery algorithm ST with the masked data. Recall that ST requires an average of all observation values provided for a grid as the initialization ground truth (i.e., Step 1 in Alg. 2.1), which obviously cannot be calculated with the uploaded masked data. The historical ground truth or a random value can be used as a substitution.

A threshold τ of *sufficient* or *insufficient* should be determined based on different scenarios. The final weight $w_{s,t}$ is calculated by averaging the two results of the two algorithms. The TM then appends the final weight to W_s for s by linking RID_s with PID_s, which is not shown in Fig. 2.1.

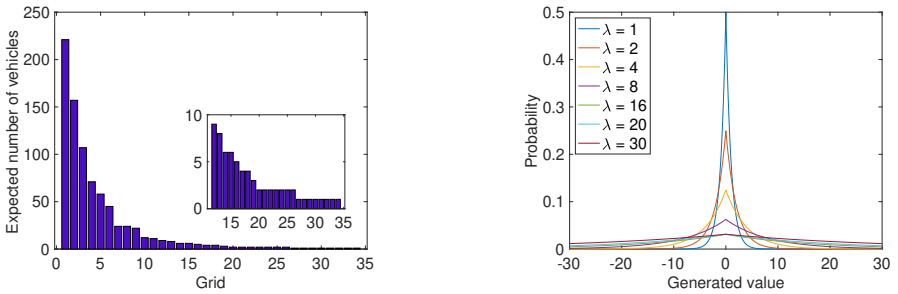
2.6 Performance Evaluation

2.6.1 Truth discovery performance

In this section, we conduct simulations to evaluate the performance of truth discovery EAirQ. A common and widely accepted truth discovery algorithm introduced in [74] is simulated for comparison, denoted as **TD** (**T**ruth **D**iscovery) in the following. We also compare EAirQ with our previous work [63] (called **AirQ**), where only the ST algorithm is used.

Simulation setup

We first introduce the simulation setup for evaluating the performance of truth discovery as follows:



(a) A Zipf distribution example

(b) Laplace distributions with different λ

Figure 2.2: Zipf and Laplace distributions

Dataset of grids and truths: we adopt a dataset containing the Air Quality Index (AQI) from 34 base stations in January 2020 in Beijing, China [10]. Each base station in the dataset is regarded as a grid. The geographical distances among grids are calculated with the longitudes and latitudes of the base stations. The AQI values are used as original real truths. We observe that there is nearly

no temporal correlation because of the coarse granularity of record periods and grids. Thus, we interpolate three evenly spaced values between every two AQI values (i.e., in every hour). As a result, the sensing cycle is 15 minutes and there are 2973 real truths in total for each grid. We use $\hat{v_g}$ to denote the real truth of grid g.

Long tail phenomenon: in the dataset, some base stations (e.g., Qianmen) are located in busy commercial centers while some (e.g., Yungang) out of the Fourth Ring Road of Beijing, i.e., not in busy areas. This intuitively leads to an obvious difference in vehicle densities, which is similar to the observation in [117], i.e., the long tail phenomenon. To simulate the phenomenon, we set the expected number of vehicles passing by a grid following a Zipf distribution. In other words, in a sensing cycle, most of the vehicles are expected to pass by a small portion of grids in our simulations. One example is shown in Fig. 2.2a. Note that we consider each sensing cycle is independent so that the expected number is used as the mean value in a Poisson distribution to generate the exact number of vehicles passing by the grid in the cycle.

Vehicle trajectories: to simplify the simulation, instead of generating a real trajectory for a vehicle, we only set up a "simplified trajectory" in advance, without considering the sequence of the vehicle movements. To be specific, in each sensing cycle, we randomly generate a list of vehicles for each grid g. The length of the list is the exact number of vehicles passing by g, i.e., the number generated above with the Poisson and Zipf distributions. An example is given in Fig. 2.3. The red check mark represents s_1 passes by g_1 in the sensing cycle 1 and contributes an observation value. s_1 also passes g_m but does not pass g_2 in this cycle. We assume that each vehicle only contributes zero (cross mark in Fig. 2.3) or one (check mark in Fig. 2.3) observation value for one grid in each sensing cycle, as mentioned in Section 2.4.

Vehicle reliability: considering the different precisions of onboard sensors and the possibility of malicious vehicles, a reliability value should be initialized for each vehicle in advance. In our simulations, we set a deviation value κ_s following a truncated Normal distribution $\mathcal{TN}$ to represent the reliability. Recall that the reliability of sources is unknown a priori in practice. The initialized reliability in the simulations is only used to generate corresponding observation values

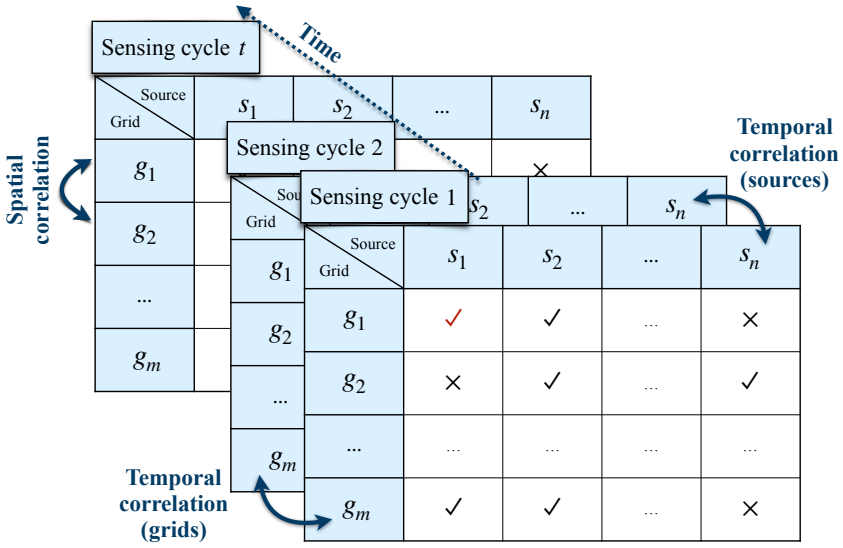


Figure 2.3: Observation values generation

and never accessible to the server.

Observation values: as a systematic bias, κ_s can be used as a multiplicative factor of each real truth $\hat{v_g}$ to generate the observation value for s. To be specific, $\hat{v_g} \cdot \kappa_s$ is the expected observation value for grid g provided by s. We adopt a Normal distribution $\mathcal{N}$ to simulate the accidental bias. We choose the variance of the distribution as 0.2 so that the accidental bias has low impacts. Supposing $g^{s,j} = g$, the observation value $v_{s,j}$ is generated from $\mathcal{N}(\hat{v_g} \cdot \kappa_s, 0.2)$.

Evaluation metrics: we evaluate the performance of truth discovery based on two metrics.

- **Root-Mean-Square Error (RMSE)**: the average root-mean-square deviation of each sensing cycle. To make figures clear, we calculate the average RMSEs for each day, i.e., 96 sensing cycles;
- **Valid Estimations**: the number of records, i.e., estimated ground truths, whose relative deviations from the real truths are less than a threshold.

Truth discovery performance of EAirQ

We first discuss the introduced parameters in this section. Then we compare the performance of TD, AirQ and EAirQ with good sources. In addition, we analyze the

impact of the perturbation mechanism on the accuracy of truth finding. Finally, we compare the performance of TD, AirQ and EAirQ with bad sources.

There are seven important parameters we introduced in the simulations:

Source reliability κ_s: the source reliability $\kappa_s \sim \mathcal{TN}(1.5, 0.5, 1, \sigma)$ where the parameters of the truncated Normal distribution are the upper limit, the lower limit, the mean and the standard deviation, respectively. Obviously, now all the observation values are in an acceptable range of $[0.5\hat{v_g}, 1.5\hat{v_g}]$. σ are set differently to simulate the scenarios where most of the vehicles are normal ($\sigma = 0.5$), a great number of vehicles are normal ($\sigma = 1$), and a great number of vehicles are abnormal ($\sigma = 2$). Note that although we use "normal" and "abnormal" to describe the sources with different reliability, all the sources with this setting are *good*. In other words, these sources may have different κ_s because of varying sensor precisions but the mean value is 1. It is reasonable and common in practice.

Threshold for valid estimations: because there is no standardized threshold in related works, we intuitively set the thresholds as 15%, 20% and 25% for different error-tolerant levels. The reason is that the air pollution categories are defined by every 50 or 100 scores of AQI according to the Technical Regulation on Ambient Air Quality Index [75]. For example, AQI among 0–50 represents the category of *Excellent* and 200–300 represents *Heavily polluted*. A deviation of 15%, 20% or 25% is acceptable considering the category step.

Probability p_1: the probability with which a vehicle removes an observation value from the report in the grid perturbation process, as defined in (2.26). Recall that we generate a list of vehicles for each grid based on a Zipf distribution. We observed that, under the simulation settings, the total number of observation values provided by a vehicle (i.e., the number of grids passed by) in a sensing cycle usually has a mean less than 10. It corresponds with the practical situation: a vehicle usually cannot travel the majority of grids in one cycle. With this observation, one acceptable value of p_1 is 0.2. Then the expected number of removed observation values for a vehicle in a sensing cycle is less than 2.

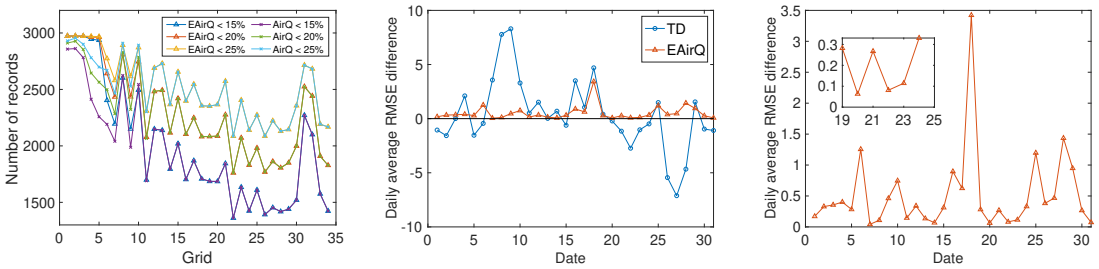
Probability p_2: the probability with which a vehicle generates an observation value for a grid it does not pass by in one sensing cycle in the grid perturbation process. As mentioned in Section 2.5.4, we suggest setting a small value for p_2

considering the accuracy of truth discovery. For example, if $p_2 = 0.05$, recall that there are 31 grids in total and the number of grids a vehicle s passes by is usually less than 10, the expected number of simulated observation values is more than $(31 - 10) \times 0.05 = 1.05$ for s.

Scale parameter λ_1: the scale parameter for the Laplace distribution $\mathcal{L}(0, \lambda_1)$ which is used to generate ψ_1 in the grid perturbation process. Because the AQI usually ranges from 20 to 100 in the dataset we adopted, adding a single-digit noise is acceptable. The probability density functions of the Laplace distributions with different scale parameters are shown in Fig. 2.2b. Thus, intuitively, λ_1 is better to be set around 2.

Scale parameter λ_2: the scale parameter for the Laplace distribution $\mathcal{L}(0, \lambda_2)$ which is used to generate ψ_2 in the value perturbation process. Similar to λ_1, setting λ_2 to around 2 is reasonable. Because every value perturbed with λ_1 is expected to be perturbed again with λ_2, we suggest setting λ_1 smaller than λ_2. Intuitively, we set $\lambda_1 = 1.5$ and $\lambda_2 = 2$ in our simulation.

Report threshold τ: the threshold number of reports for a grid in a sensing cycle, which is used in the data handling process. Considering both the simulation results analyzed in the previous work [63] and the Zipf distribution shown in Fig 2.2a, we set τ to 10, which is approximately the dividing line between grids 1 to 11 and 12 to 34.



(a) Valid estimations

(b) RMSE differences between TD/EAirQ and AirQ

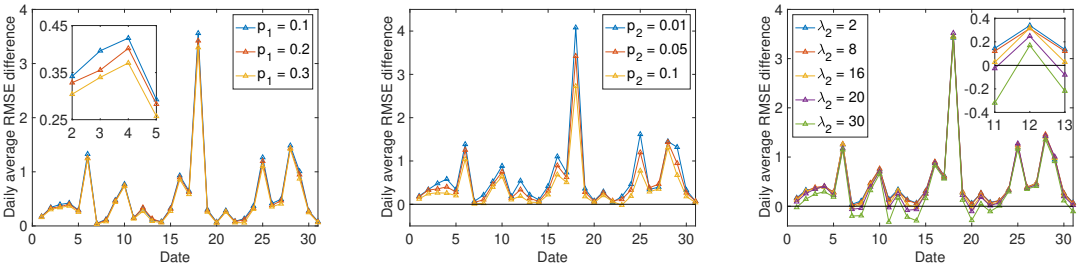
(c) RMSE differences between EAirQ and AirQ

Figure 2.4: Performance of EAirQ with good sources

We first conduct simulations on 500 good vehicles with $\sigma = 0.5$, $p_1 = 0.2$, $p_2 = 0.05$, $\lambda_1 = 1.5$, $\lambda_2 = 2$ and $\tau = 10$. The valid estimations are shown in Fig. 2.4a.

Obviously, AirQ and EAirQ perform nearly the same for grid 12 to 34, which echoes the design of EAirQ, i.e., to use the ST algorithm for grids with sufficient data. What we concern about is the grids 1 to 11. It shows that EAirQ has more valid estimations than AirQ.

To get clearer observations, we introduce a new evaluation metric based on the daily RMSE: *Daily RMSE Difference*. It is the difference between two daily RMSEs. In our simulations, we always use the RMSEs of AirQ to subtract that of TD or EAirQ. Thus, if the difference is larger than 0, we can say TD or EAirQ works better than AirQ. The comparison results are shown in Figs. 2.4b and 2.4c. It is clear that EAirQ always has a positive difference value, which shows that EAirQ works better than AirQ.



(a) Different p_1 (b) Different p_2 (c) Different λ_2

Figure 2.5: RMSE differences between EAirQ and AirQ with different perturbation parameters

Table 2.2: Parameter settings for comparison

Results	Parameters			
	p_1	p_2	λ_1	λ_2
Fig. 2.5a	0.1, 0.2 or 0.3	0.05	1.5	2
Fig. 2.5b	0.2	0.01, 0.05 or 0.1	1.5	2
Fig. 2.5c	0.2	0.05	1.5	2, 8, 16, 20 or 30

Fig. 2.5 shows the impact of different perturbation parameters. The settings of parameters are listed in Table 2.2. Note that because both λ_1 and λ_2 are used to generate the Laplace noise and λ_2 affects much more data than λ_1, we only simulate with different λ_2 as an instance. We can observe that: a) a smaller p_1 or p_2 results in a higher RMSE difference. In other words, the estimated ground truths are more

accurate; b) the impact of p_2 is more significant than p_1 because a larger p_2 introduces more imitated reports. When $p_2 = 0.1$, EAirQ even performs a bit worse than AirQ on the 23rd day; c) a higher λ_2 leads to worse performance. Negative difference values are observed if $\lambda_2 > 16$. These results are reasonable because a higher p_1, p_2 or λ_2 leads to more noise on the reports. Controlling the bias in an acceptable range and finding a balance between privacy and accuracy can be achieved by adjusting the parameters. Besides, an interesting observation in Fig. 2.5 is that, the results with $\lambda_2 = 2$ and $\lambda_2 = 8$ are close to each other. It is because we only use the perturbed data when there are sufficient reports and the symmetric Laplace noise can be canceled to some extent. In practice, λ_2 can be set a bit larger than 2 but the issues of weight management should be considered. We discuss more in Section 2.7.

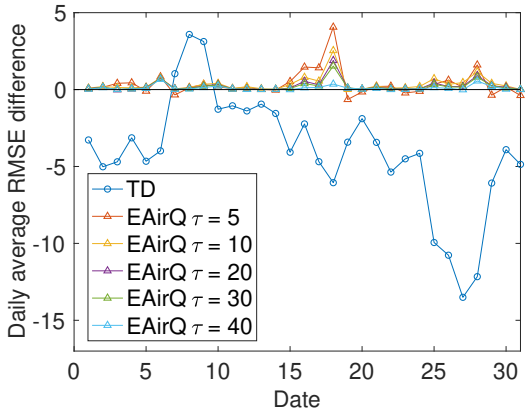


Figure 2.6: Performance of EAirQ with 15% bad sources

Based on what we observed from the previous work [63], 0% to 10% bad sources do not make much difference under the simulation settings. Thus, we conduct simulations under the scenario with 15% bad sources to evaluate the performance of EAirQ further. The simulations still use the setting that $\sigma = 0.5$, $p_1 = 0.2$, $p_2 = 0.05$, $\lambda_1 = 1.5$, $\lambda_2 = 2$ and $\tau = 10$. Results are shown in Fig. 2.6. It can be seen that, EAirQ has higher accuracy than TD and AirQ. One point worth mentioning is that the choice of threshold τ is important but complex. To be specific, the relationship between τ and the truth discovery accuracy is not monotonically increasing or decreasing. For example, although $\tau = 10$ leads to a higher RMSE difference than $\tau = 40$ in most of the cases, $\tau = 5$ sometimes even leads to a negative difference value. Under our simulation settings, $\tau = 10$ is an acceptable choice.

2.6.2 Computation cost

Table 2.3: Comparison on the computation cost

Framework	Vehicles			Cloud server
	Additions	Multiplications	Exponentiations	
EAirQ	$\mathcal{O}(c^2m)$	$\mathcal{O}(cm)$	0	No additional costs
PPTD [73]	$\mathcal{O}(rm)$	$\mathcal{O}(rm)$	$\mathcal{O}(rm)$	Additional costs

Considering that there is no privacy-preserving mechanism in TD, we introduce an extended scheme of TD, **PPTD** [73] for comparison. PPTD adopts the Threshold Paillier cryptosystem to protect user privacy. The computation cost of EAirQ is analyzed and compared with that of PPTD in Table 2.3.

In EAirQ, a vehicle first calculates $\{\theta_{s,j,g} \cdot v_{s,j} \mid j \in \{1, 2, \ldots, c\}\}$ and $\{\theta_{s,j,g} \cdot v_{s,j}^2 \mid j \in \{1, 2, \ldots, c\}\}$ for g as a preparation of the masking process. Thus, there are $2c$ multiplications. Then, $c-1$ additions are needed to mask each value by (2.20). For all the $3c$ values, i.e., $\{\theta_{s,j,g} \cdot v_{s,j} \mid j \in \{1, 2, \ldots, c\}\}$, $\{\theta_{s,j,g} \cdot v_{s,j}^2 \mid j \in \{1, 2, \ldots, c\}\}$ and $\{\theta_{s,j,g} \mid j \in \{1, 2, \ldots, c\}\}$, $3c(c-1)$ additions are needed for the masking mechanism. Besides, a vehicle s needs to add noise to the observation values. As described in Section 2.5.4, new observation values are expected to be generated and perturbed for the $m-c$ unpassed grids with a probability p_2 by (2.27) and then all the observation values will be perturbed by (2.28). Thus, there are $p_2(m-c)+c$ addition operations for s. A strength of EAirQ is that vehicles do not need to participate in the following processes after uploading the masked and perturbed data. Overall, a vehicle calculates $\mathcal{O}(c^2m)$ additions and $\mathcal{O}(cm)$ multiplications in EAirQ where m is the number of grids.

Note that the computation cost of the anonymous authentication is not included in this section because there is no specific signing and verification scheme adopted in PPTD. From the perspective of security, any messages sent should be signed and encrypted so that it is necessary to introduce an authentication scheme in PPTD in practice. Based on the analysis in [77], the cost of signing is acceptable in EAirQ.

In PPTD, there are multiple updating rounds for truth discovery until a convergence criterion is satisfied, which is similar to the case in EAirQ. However, the sources should participate in every round to fulfill the truth discovery in PPTD. (Re-

call that a source is a vehicle in our work but PPTD is not designed particularly for vehicular networks. Thus, we prefer to use the term "source" instead of "vehicle" when discussing PPTD.) In each round, some preparation operations, threshold encryption and decryption are needed. Because the processes are complex, we give the computation cost directly: $\mathcal{O}(mr)$ additions, $\mathcal{O}(mr)$ exponentiations and $\mathcal{O}(mr)$ multiplications for each source where r is number of updating rounds. Note that m is the number of grids but also the number of reports in PPTD because a source is assumed to provide observation values for all grids in PPTD. Some operations are in different fields (such as a multiplicative group) in PPTD but we do not distinguish them in this chapter for simplicity. Readers can refer to [73] for more details.

With regard to the cost of the server, we only give a brief discussion, since the server is expected to be configured on cloud with sufficient calculation resources. As shown in Table 2.3, except the necessary calculations for truth discovery, there is no additional computation costs in EAirQ. In other words, there is no need to remove the randomness from the masked or perturbed values. The server can update the truths and weights based on (2.24), (2.25), (2.18) and (2.19) directly. However, in PPTD, the server should perform some multiplications and exponentiations to handle the ciphertexts first before using them in the truth and weight updating.

Overall, EAirQ is lightweight, especially on the vehicle-side.

2.6.3 Communication cost

We analyze and compare the communication costs in three aspects as follows:

Communication rounds: in PPTD, there are $\mathcal{O}(r)$ updating rounds for truth discovery in one sensing cycle. In each round, a source should communicate with the server at least two times. The messages sent in one round are $\mathcal{O}(m)$ ciphertexts. Thus, a source sends $\mathcal{O}(rm)$ ciphertexts in each sensing cycle. The size of a ciphertext depends on the parameters chosen for the threshold cryptosystem.

In EAirQ, there are $c + 1$ communication rounds in total in each sensing cycle. In each of the c rounds, a vehicle requests $\mathcal{O}(m)$ parameters from an RSU. In the last round, the vehicle uploads the report containing $\mathcal{O}(cm)$ masked or perturbed values.

Overall, suppose the cost of sending one value (a ciphertext, a parameter, a masked value or a perturbed value) is $\mathcal{O}(1)$. Then the communication cost

for a source is $\mathcal{O}(rm)$ in PPTD and $\mathcal{O}(cm)$ in EAirQ. Considering that the communication technologies of vehicular networks are improving fast, it should not be a bottleneck.

Communication architectures: EAirQ is more suitable for VANETs from the perspective of communication architectures. To be specific, sources in PPTD need to communicate with the server directly which results in a long communication time. In EAirQ, RSUs are intermediaries to forward the messages between vehicles and the server. Thus, vehicles do not need to wait for the reply from the server, which is an advantage of EAirQ.

Traffic bursts: in PPTD, a source needs to conduct the $\mathcal{O}(r)$ communication rounds in a short period when the server asks for data updating and truth discovery. However, the $c+1$ communication rounds of EAirQ are scattered in the whole sensing cycle. Only the last round is for data updating. Thus, it will not lead to bursts and high overhead for the communication network.

2.6.4 Privacy analysis

To protect the privacy, we adopt a masking algorithm for ST. We first summarize the security of the masking algorithm: a) the construct of the mask hides all information of individual inputs except for their sum; b) the PRNG algorithm provides *pseudo-randomness* to the chosen values (or masks); c) the masks are used as one-time pads, which provides *true randomness* for the algorithm. Besides, we present a two-layer perturbation mechanism for SST and adopt an anonymous communication scheme in EAirQ.

The masking mechanism and the perturbation mechanism protects the privacy of observation values and vehicle trajectories as follows:

Privacy of observation values: with the masking mechanism, in the process of data uploading, each observation value $v_{s,j}$ is not sent directly, as described in Section 2.5.3. In the data handling process, the cloud server can obtain the sums of $\{\theta_{s,j,g} \cdot v_{s,j} \mid j \in \{1, 2, \ldots, c\}\}$ and $\{\theta_{s,j,g} \cdot v_{s,j}^2 \mid j \in \{1, 2, \ldots, c\}\}$ by summing the masked values, $\{\beta_1^{s,j,g} \mid j \in \{1, 2, \ldots, c\}\}$ and $\{\beta_2^{s,j,g} \mid j \in \{1, 2, \ldots, c\}\}$, respectively. In other words, featured by the masking technique, there is no need for the server to acquire the masks (i.e., the chosen random values) or any $v_{s,j}$.

In addition, the masks used in constructing $\{\beta_1^{s,j,g} \mid j \in \{1,2,\ldots,c\}\}$ and $\{\beta_2^{s,j,g} \mid j \in \{1,2,\ldots,c\}\}$ are chosen and only known by the vehicle s. They work as one-time pads, i.e., they are never reused in other sensing cycles. Thus, it is infeasible for any other parties except s to remove the randomness and infer any value of V_s.

With the perturbation mechanism, every observation value is first perturbed locally by the vehicle s with a Laplace noise before uploading. It is infeasible to guess the exact value of the noise. Thus, the true observation values are never disclosed to any parties except s.

Privacy of vehicle trajectories: with the masking mechanism, the server only use (2.24) and (2.25) for truth discovery. It does not need to know the exact information of the locations of each report in addition. Then, the only parameter that may disclose the information of locations is $\theta_{s,j,g}$. It represents the logical distance of grids $g^{s,j}$ and g. However, all $\{\theta_{s,j,g} \mid j \in \{1,2,\ldots,c\}\}$ are masked to $\{\beta_2^{s,j,g} \mid j \in \{1,2,\ldots,c\}\}$ and $\{\beta_3^{s,j,g} \mid j \in \{1,2,\ldots,c\}\}$ for each grid g by source s. The masked values are sent to the server in reports and used to calculate the sums needed. It is infeasible for any other parties except s to obtain the exact value of any $\theta_{s,j,g}$ from the reports or the sums. Thus, the privacy of trajectories is preserved.

With the perturbation mechanism, every grid that a vehicle s passed by is expected to be concealed with p_1. It is infeasible for any parties except s to know which trajectory is removed. Besides, $(m-c)p_2$ imitated observation values are expected to be generated where c is the number of observation values provided before the perturbation. Thus, each location inferred from the report has a probability of $\frac{(m-c)p_2}{(m-c)p_2+c}$ to be false. Recall that a Laplace noise is added to the imitated observation values by (2.27), which makes it challenging to judge if a value is manually generated from the historical truth. In other words, it is challenging to distinguish a false trajectory from the true trajectory.

As for the anonymous communication, the privacy properties of it are as follows: a) a vehicle s never uses its real ID RID_s to upload reports. Both the RSUs and the server cannot infer the true identity of s; b) s can use any of the many existing pseudonyms changing techniques to change its pseudonym among sensing cycles [77]; c) only the TM knows the RID_s. When the server requires the historical weights

for s with the pseudo-ID PID_s from the TM, only the weights are returned; and d) the weight histories maintained by the TM are expected to be changing based on the application-layer user activities, which increases the challenges to link a record with a vehicle by the server. The four properties guarantee that it is infeasible for the server to infer the relationship between a report with a vehicle. In addition, the RSUs cannot trace a vehicle in the periodical communications. Thus, both the observation values and the trajectories are protected.

Note that EAirQ preserves the privacy of observation values and vehicle trajectories while PPTD preserves that of observation values and weights. The difference in privacy goals is reasonable considering the practical scenarios and applications.

2.7 Discussion

Possible scenarios: vehicles with high mobility can collect various data from the environment, such as the noise level, the humidity, the temperature, and the flow density. Thus, EAirQ is not restricted to air quality monitoring. It can also adapt to other scenarios well. The remaining work is to adjust the parameters based on the degrees of temporal and spatial correlations. For example, we can decrease ρ_t in (2.8) properly for the sensing task with a significant temporal correlation on ground truths such as the outdoor temperature. Similarly, the threshold u and width parameter ω in (2.3) should be adjusted based on the spatial property of the sensing object. To emphasize that, although the parameters are expected to be changed for different applications, we do not need to modify the formulas for truth discovery.

A possible extension of correlations: in this work, only the temporal and spatial correlations are involved, but we discuss the possibility of the *attributes-based correlation*. Intuitively, two grids are likely to have similar air quality values if they are similar in some particular attributes. For instance, the construction sites with heavy-duty engines are more likely to produce diesel emissions.

We define the similarity of $g^{s,t}$ and g as:

$$\mathrm{Sim}(g^{s,j}, g) = \begin{cases} 1 - \mathrm{Nor}(\mathrm{Lis}(L_{g^{s,j}}, L_g)), & \text{if } \mathrm{Nor}(\mathrm{Lis}(L_{g^{s,j}}, L_g)) < u' \\ 0, & \text{otherwise} \end{cases} \tag{2.29}$$

where $L_g = \{f_1, f_2, \ldots, f_z\}$ is the vector of z attributes of grid g, including

the vehicle density, the vegetation ratio, the number of factories, the duration of constructions and so on. Similarly, $L_{g^{s,j}} = \{f'_1, f'_2, \ldots, f'_z\}$ is the attribute vector of $g^{s,t}$. $\mathrm{Lis}(L_{g^{s,j}}, L_g)$ is the *Lance and Williams distance* of two vectors $L_{g^{s,j}}$ and L_g. The Lance and Williams distance is also called *Canberra distance*, which assumes that the variables in a vector are independent of each other. It is widely used to measure the similarity and the dissimilarity between groups. To be specific, $\mathrm{Lis}(L_{g^{s,j}}, L_g)$ can be calculated as follows:

$$\mathrm{Lis}(L_{g^{s,j}}, L_g) = \sum_{i=1}^{z} \frac{|f'_i - f_i|}{|f'_i| + |f_i|}. \tag{2.30}$$

Recall that the attributes of all the grids can be known in advance by the RSUs. Thus, the distances between the attribute vectors can be calculated and normalized by RSUs in advance as well. Note that the normalization is processed among all the distances but we use $\mathrm{Nor}(\mathrm{Lis}(L_{g^{s,j}}, L_g))$ in (2.29) to denote the normalization result of $\mathrm{Lis}(L_{g^{s,j}}, L_g)$. $\mathrm{Nor}(\mathrm{Lis}(L_{g^{s,j}}, L_g)) \in [0, 1]$. If two grids are similar, they are expected to have a small Lance and Williams distance and the normalization result tends to 0. Thus, the similarity is finally defined as $1 - \mathrm{Nor}(\mathrm{Lis}(L_{g^{s,j}}, L_g))$ when a distance threshold u' is satisfied.

The parameter $\theta_{s,j,g}$ then can be redefined as

$$\theta_{s,j,g} = \psi_1 \,\mathrm{Dis}(g^{s,j}, g) + \psi_2 \,\mathrm{Sim}(g^{s,j}, g) \tag{2.31}$$

where ψ_1 and ψ_2 control the weights of the two correlations. The challenge of adopting the attributes-based correlation is the complexity of attributes.

The trade-off between accuracy and privacy: we have mentioned that adding more noise to reports can provide a higher privacy but may lead to a lower accuracy. A system manager can weigh it according to specific demands. For example, a system that needs results of concrete values should have less noise added than a system that only needs classified outputs (such as *Heavily polluted*).

The parameters p_1, p_2, λ_1 and λ_2 can be adjusted accordingly. As observed from Fig. 2.5c, λ_2 can be set larger than what we use in most of the simulations (i.e., 2). It provides more privacy but does not affect the accuracy much, which is the strength of the Laplace noise. However, one point needed to discuss is the

weight issues introduced by it. With more noise added to the observation values, the estimated weights for sources are deflected more from the real reliabilities. If the estimated weights are only used for truth discovery, it is acceptable to use a larger λ_2. If the system manager uses the weights for some application-layer functions as well, such as giving rewards to sources based on their weights, we suggest keeping λ_2 small.

Weight management: recall that the TM has the ability to update the historical records of weights so that the changing records provide better protection of the privacy. However, because it is not the main focus of our work, we did not discuss much and there are many remaining problems. For example, how to define a bad source with the weight? In other words, what should be the threshold of the weight for a bad source? How to reasonably update the whole W_s for each s rather than only the latest $w_{s,t-1}$ by the TM? A concrete list of updating and management rules should be proposed in the future.

Datasets and simulations: the simulations conducted in this work are not perfect. We adopt a dataset and manually generate some data such as the observation values. The dataset is the most fine-grained one that we can find. However, it is not enough. Getting a dataset with the truths of streets or blocks is challenging in practice. We believe the limitation of datasets affects the performance shown in the simulations. Besides, if it is possible to conduct experiments with sufficient volunteers in practice, a more convincing evaluation can be provided. Based on our observations, the discussed challenges are common in similar crowdsensing research.

Vehicle mobility: recall that we set up simplified trajectories for vehicles, instead of generating real trajectories with concrete mobility settings. The reasons that we do not adopt a concrete mobility model in this work are twofold: 1) without considering the communication performance (such as the packet loss rate), the vehicle mobility does not affect the truth discovery performance. Vehicles only upload the reports to the nearest RSU in each sensing cycle. After that, the vehicles do not need to participate in any following data handling process. Thus, they can travel to any destination at any speed. 2) Definitely the vehicle movements have an influence on the communication performance in VANETs. However, the corresponding QoS is not the main focus of this work, as we as-

sumed in Section 2.4. As a future work, some complex mobility models (such as that in [88]) and the effects on the QoS can be taken into consideration. Besides, whether and how the mobility, especially the speed, affects the precision of onboard sensors is also a potential research topic.

2.8 Conclusion

we presented a truth discovery algorithm for vehicular crowdsensing incorporating the spatial and temporal correlations. We further proposed a lightweight privacy-preserving framework, EAirQ, based on data masking, anonymous communication and perturbation. Two different truth discovery algorithms are used in EAirQ. EAirQ can address the data sparsity problem and preserve the privacy of reports and trajectories at the same time. We conducted simulations to measure the performance and analyzed the privacy achievements of the framework. Results show that EAirQ protects the privacy, has a good truth discovery performance, and keeps the computation and communication costs low.

The previous work of EAirQ, i.e., AirQ, is not included in this dissertation considering the limit of length. Readers can refer to [63] for more details. The typical limitations discussed in AirQ have been addressed in this chapter.

To address the remaining limitation related to vehicle mobility, we have conducted evaluations with real mobility models in our next work, introduced in Chapter 3. Other limitations are open questions or out of the scope of the dissertation, and therefore we have not worked further on them.

Chapter 3

CRS: Two-layered Privacy-preserving Distributed Machine Learning

In Chapter 2, we proposed a privacy-preserving truth discovery framework for data aggregation in the Internet of Vehicles (IoV), which mainly targets the scenario of fine-grained air quality monitoring. This framework utilizes an end-to-end architecture in IoV, i.e., individual data is uploaded directly to the server by each vehicle via road-side units (RSUs). A question arises in our mind: how can we design a specialized data aggregation architecture for IoV with fully considering the characteristics of vehicular ad-hoc networks? In this chapter, we present our efforts to address this question: we propose a privacy-preserving two-layered architecture with vehicle clusters. Considering the rapid development of artificial intelligence (AI) and the wide range of potential applications, we primarily focus on the scenario of distributed machine learning in this chapter but it is important to note that the proposed architecture definitely can adapt to general sensory data aggregation scenarios.

To be specific, in this chapter, we present a two-layered distributed machine learning framework. The architecture uniquely involves vehicle clusters, RSUs, and a central server, which provides a basic guarantee to vehicle privacy and also limits the overhead. By carefully adopting cryptographic tools and techniques, the framework has the following properties: a) it preserves the privacy of the local inputs and model weight vectors to all parties; b) it protects the identities and trajectories of vehicles; c) packet loss is handled in the application layer; d) the evaluation shows that it is

lightweight for vehicles. Compared with other existing works, the proposed framework is more suitable for IoV.

The frequently used notations in this chapter are summarized in Table 3.1 for reference.

Table 3.1: Notations in Chapter 3

Notation	Description
CS	The cloud server
$\{v_i \mid i \in N_{\mathrm{v}}\}$	A cluster composed with n vehicles. $N_{\mathrm{v}} = \{1, 2, \ldots, n_{\mathrm{v}}\}$
CH	A cluster head
(X_i, Y_i)	Video recordings and the corresponding experience kept by v_i
ρ_i	The weight of vehicle v_i for head election
$\alpha_{i,j}$	The mask agreed between v_i and v_j
w_i	The local model weight calculated by v_i
w_{local}	An aggregated cluster model weight
w'	A temporary global model weight
w	The global model weight
c_i	The masked value of w_i
β	Parameters to generate linear equations for message encoding
$\mathrm{COD}(i\|\|j)$	The encoded message for c_j generated by v_i
$pk = (pk_1, pk_2)$	The public key pair of CS used in the Paillier encryption
pk'	The public key of CS used in the ElGamal encryption
pk_{RSU}	The public key of the RSU
n_{c}	The number of clusters
n_{r}	The number of video recordings a vehicle keeps *
n_{f}	The number of frames in a video *
$x_{i,j}$	The information extracted from a video recording $X_{i,j}$ *
std	The standard deviation used to generate vehicle speeds in the emulated scenario

* *Only used in the application instance.*

3.1 Introduction

Nowadays, many vehicles are equipped with an on-board digital video recorder (i.e., dashcam). The videos play an increasingly important role today with the technique of machine learning [18, 100]. For example, with the information extracted from the video, vehicles can make real-time driving decisions in automatic driving. The traffic accidents and the severity can also be detected from the videos. To make these decisions and detections accurately, a proper machine learning model is required to be trained with a large volume of data first. However, traditional machine learning approaches have significant limitations. The training data are generated from numerous vehicles all the time. It is hard to gather all the data to the server efficiently.

The rapid growth of Internet of Vehicles (IoV) offers a promising solution for this problem. Different from the traditional vehicular ad-hoc networks (VANETs), IoV is formed by vehicles maintaining not only the communication capability but also their own storage, computing, and learning capabilities. Thus, vehicles can perform as local workers in distributed machine learning (DML) [1]. DML allows multiple vehicles to carry out the stochastic gradient descent (SGD) at their locality with disjoint training data shards. The server only needs to aggregate the local training results. Because there is no need to transmit the local datasets, DML not only reduces the communication cost, but also preserves the privacy of the training data.

Many studies focusing on vehicular machine learning have been conducted in recent years [3, 27, 102, 94]. The approaches usually aim at generating more reliable classification or decision models. However, some significant issues need to be well addressed. First, the architecture of a distributed learning framework has a considerable impact on efficiency, which is even more obvious in IoV. To be specific, the architecture includes how to deploy vehicles, how the vehicles communicate with the server, and so on. Another crucial problem is the leak of privacy. The privacy is not restricted to the contents of the videos, but also the local training results which may expose information on the raw data [84]. Further, the identities and trajectories of vehicles are also important in IoV.

To address the above concerns, we propose a distributed machine learning framework for IoV, named **CRS** (**C**lusters-**R**SUs-**S**erver). It has a novel two-layered ar-

[1]DML might seem similar to federated learning (FL). FL is a form of DML but a) emphasizes more on privacy protection, and b) is faced with a more complex environment [122]. For example, the training data in FL is usually considered to be non-independent and identically distributed (non-IID).

chitecture, which has natural advantages over traditional end-to-end architectures: both the overhead of vehicles is reduced and the privacy is preserved. In CRS, each vehicle generating video recordings is not required to upload the recordings. Instead, it trains a gradient or model weight locally. Clusters are formed through vehicles and compose the first layer with road-side units (RSUs). Local gradients or model weights are first aggregated by a cluster head in a secure way with network coding and masking. RSUs and the server compose the second layer, where intermediate gradients or model weights are transmitted with a threshold homomorphic cryptography system to get a global gradient or model weight. Simulations have been conducted in ns-3 (Network Simulator-3) [80] with both emulated and real-world datasets. The results show that the communication and computation costs are relatively low for vehicles and acceptable for the entire vehicular network. We also provide use cases and a detailed application instance of applying CRS in a concrete problem.

The main work and contributions are as follows:

- We extract a simplified setup protocol from an existing multi-hop vehicle clustering algorithm. Not only a cluster head can be elected but also secret parameters can be exchanged among vehicles. The protocol can be combined with any strategies to meet the different requirements of the cluster head election.

- We propose a secure and reliable data aggregation protocol for vehicle clusters, based on the network coding and the masking techniques. The sensitive data provided by each vehicle will not be released while the representative vehicle can aggregate all the data with acceptable communication and computation costs. Based on the non-equal protection (NEP) theory, the loss of critical messages is also considered and handled in the process.

- Different from traditional end-to-end distributed machine learning frameworks, the proposed CRS has a novel "vehicle clusters-RSUs-server" architecture. All entities are deployed uniquely in the two-layered architecture to take full advantage of the properties of clusters and IoV. Furthermore, in CRS, the proposed secure and reliable data aggregation protocol and a threshold homomorphic cryptography system work together to provide confidentiality and preserve privacy.

- The proposed CRS is more suitable for IoV on the following three aspects: a) the accuracy of existing machine learning frameworks usually relies on the as-

sumption of no packet loss, which is impractical in vehicular networks. CRS handles packet loss in the application layer. b) Existing works focus on the protection of local data. In our work, besides that, the identities and trajectories of vehicles are not revealed to the server. c) The burden is shifting from individual vehiclesto RSUs and the server. Thus, CRS is more lightweight for vehicles.

To the best of our knowledge, this is the first work that aims at designing a machine learning framework specifically for IoV, where the data privacy, the identity privacy, the trajectory privacy, the packet loss, and the communication and computation costs of vehicles are all considered.

In the rest of the chapter, we show the related work in Section 3.2, introduce the adopted techniques in Section 3.3, and define the problem and the threat model in Section 3.4. The proposed framework is introduced in details in Section 3.5. We provide a concrete application instance of CRS and discuss some use cases and the prospects in Section 3.6. In Section 3.7, we evaluate the performance of the framework and analyze the privacy. Possible future work and conclusion are given in Sections 3.8 and 3.9.

3.2 Related Work

3.2.1 Privacy preservation in machine learning

There are many existing works focus on the privacy and security of machine learning. Shokri *et al.* [99] design a practical deep learning system. In this system, workers train independently on their own datasets. Instead of sharing the datasets with the server, they selectively send the key local training results. Compared with the centralized machine learning, this work achieves much stronger privacy and nearly the same model accuracy. However, Phone *et al.* [84] claim that an adversary can infer the raw datasets with a small portion of the local training results. Thus, protecting the local training results is also crucial for distributed machine learning.

To preserve the privacy of local training results, one typical technique adopted is differential privacy [42, 90, 44]. Raja *et al.* [90] propose a distributed machine learning-based intrusion detection system, named SP-CIDS, for VANETs. Vehicles collaborate to create a global classifier to distinguish normal behaviors from intrusions. With SP-CIDS, the intermediate training results are protected by adding Laplace noise. Hu *et*

al. [42] propose a DML scheme for collaboratively training multiple personalized machine learning models. The Gaussian mechanism is adopted to preserve the privacy of the local training results. Similarly, Huang *et al.* [44] also perturb the training results with Gaussian noise. They novelly combine an approximate augmented Lagrangian function with a time-varying Gaussian mechanism to achieve higher utility. However, experiments have shown that adding noise reduces the accuracy of both the shared intermediate data and the trained model [99, 90, 44]. Furthermore, Bagdasaryan *et al.* [7] demonstrate that the negative effect of differential privacy in DML is severer when the local datesets are complex, heterogeneous and underrepresented.

Another line of work adopts the homomorphic encryption algorithms [84, 41]. Phong *et al.* [84] propose a distributed deep learning system where locally trained results are encrypted first. A central server can aggregate the intermediate parameters directly without decrypting them. The system can not only preserve the privacy of the information but also keep the accuracy of the model intact. To instantiate the system, two additively homomorphic encryption schemes, the learning with errors-based (LWE-based) encryption [5] and the Paillier encryption [82], are utilized. Differently, He *et al.* [41] introduce the improved Paillier encryption scheme for DML. They compute multiple modular exponentiation operations in advance at the key generation stage to reduce the computation cost of local workers in the encryption stage. However, there are some limitations to adopt these works in vehicular networks. The communication and computation costs of vehicles should be further reduced for efficiency. Both the identities and trajectories of vehicles shall be taken into account, which are important in IoV.

3.2.2 Clustering techniques in IoV

In IoV, vehicles in a geographical vicinity can group together based on different cluster formation strategies. A *cluster head* is a representative vehicle of the cluster to exchange packets with RSUs. It can communicate with all other *member vehicle*s in the cluster.

A major concern is the stability and reliability of clusters. How to generate a stable cluster and manage clusters efficiently has been extensively studied for more than ten years. A common solution is to take the mobility of vehicles into consideration [97, 37, 91, 40, 53, 92]. Shea *et al.* [97] minimize both the distance and relative speed between the cluster head and member vehicles during cluster formation. Besides the

speed, Rawshdeh *et al.* [91] also involve mobility patterns and traveling direction. Hassanabadi *et al.* [40] form clusters using the affinity propagation algorithm where both the current position and the predicted future position of vehicles are used. Ren *et al.* [92] filter out the unstable neighbors when forming a cluster. Vehicle relative position, relative speed, and link lifetime are considered in this work. Simulations show that both the cluster head duration (i.e., the time interval from becoming a head to giving up the state) and the member duration (i.e., the time interval from joining a cluster as a member to leaving the cluster) are usually more than 160s [2].

Multi-hop clustering has become a hot topic in recent years. By allowing multi-hop communication between a member vehicle and the cluster head, we can effectively expand the cluster coverage and enhance the stability of clusters. Zhang *et al.* [126] propose a multi-hop clustering algorithm based on a priority neighbor following strategy. With setting the communication range of a vehicle to 100m, the cluster head duration is more than 110s and the member duration is more than 105s. Ucar *et al.* [110] introduce a multi-hop clustering method where new vehicles can join a cluster by directly connecting to an existing cluster member. The authors conduct simulations with realistic mobility datasets. The head duration and member duration are more than 200s and 250s, respectively, with a 200m communication range. Sivaraj *et al.* [101] focus on multi-metric head election and multi-hop intra-cluster communication. The transmission rate, the uplink/downlink channel quality and the relative distance metrics of vehicles are involved in weight calculation. A vehicle with the maximum weight is selected as the cluster head. In our work, we extract the head election process from the mechanism and modify the architecture to satisfy our setup requirements in Section 3.5.1. For more details of the mechanism, we refer readers to [101].

3.3 Cryptographic Tools and Adopted Techniques

In this section, we give a brief introduction of the cryptographic tools and techniques adopted in our work.

Data masking: data masking allows a server to aggregate data from client parties in a privacy-preserving way. An additive masking algorithm with one-time

[2]The cluster head time and member time are related to the concrete simulation settings in the corresponding works. Thus, they should not be compared directly among different works. In this section, we cite these results to show the current research and the practicability of stable clustering.

pads [15] has been adopted and introduced in Chapter 2 so that we do not repeat the details here.

Threshold Paillier cryptosystem: the Paillier cryptography system is an additive homomorphic cryptosystem. The encryption of message $m \in \mathbb{Z}_{pk_2}$ with a public key (pk_1, pk_2) is as follows:

$$\mathrm{E}_{(pk_1,pk_2)}(m) = pk_1^m r^{pk_2} \mod pk_2^2 \tag{3.1}$$

r is a random and private value that satisfies: the greatest common divisor $gcd(r, pk_2) = 1$. Suppose two random values are chosen as r_1 and r_2, the homomorphic property can be described as

$$\begin{aligned} \mathrm{E}_{(pk_1,pk_2)}(m_1 + m_2) &= pk_1^{m_1+m_2}(r_1 r_2)^{pk_2} \\ &= pk_1^{m_1} pk_1^{m_2} r_1^{pk_2} r_2^{pk_2} \\ &= \mathrm{E}_{(pk_1,pk_2)}(m_1) \cdot \mathrm{E}_{(pk_1,pk_2)}(m_2) \end{aligned} \tag{3.2}$$

In other words, the ciphertext of the sum of two messages m_1 and m_2 is the same as the multiplication of the two individual ciphertexts [74].

In our work, we adopt the secure sum protocol based on a threshold variant of the Paillier system [74]. In this protocol, the private key is divided and distributed to p parties. At least t parties should work together for decryption. We named t as the threshold of the system. Suppose each party encrypts a value with the public key of the system and sends it to Bob, representing the other party involved in the communication who wants to get the sum of all the values. Bob can first get the ciphertext of the sum by (3.2) and send it to t participants. Each participant can partially decrypt the ciphertext with the private key share. The t partial decryptions can be combined together to obtain the plaintext of the sum. More details can be found in [74].

Network coding technique: network coding is widely used to optimize the throughput of a network. For example, with a linear network coding system, each router generates linear combinations of the received packets m_i as follows:

$$m = \sum_i \beta_i \cdot m_i \tag{3.3}$$

where β_i are coefficients. Routers send the linear combinations (usually called digital evidence) rather than the same incoming packets m_i to the receiver. The receiver can recover the packets by collecting sufficient linear combinations. This breakthrough idea enhances the network throughput and constructs a more robust network.

In our work, we do not use network coding for network optimization but for message recovery in the application layer.

Anonymous authentication: the anonymity of a message sender can be provided by a digital signature-based anonymous authentication scheme. The sender uses a pseudo-ID in the signing process as a replacement for the real ID. The receiver can only verify the integrity and authenticity of the message but cannot trace the real identity of the sender. It can be achieved by a low-overhead authentication scheme proposed by Moni *et al.* [77]. In this scheme, the pseudo-IDs are issued by a trusted authority and regional trusted authorities. The signature is generated by the RSA algorithm. We refer readers to [77] for more details.

In this work, we omit the description of the parameter generation, the parameter distribution, the communication establishment and the signature construction. We assume any vehicle has the ability to communicate with pseudo-IDs. We refer to it as an *anonymous channel* for brevity.

3.4 Problem Definition and Threat Model

3.4.1 Problem definition

We denote the video recordings from vehicles as a matrix X and the corresponding experience (e.g., driving actions or decisions) as Y. Suppose there is a learning function (or a model) O_w where w is a matrix of parameters, named *model weight*. With a proper O_w, Y can be predicted from X, i.e., $Y = \mathrm{O}_w(X)$. However, O_w is usually a black box. Thus, the goal of the vehicular network is to obtain a proper and valid w by training with X and Y. In our work, the training data, i.e., X and Y, are trained locally by vehicles. A central server generates a global model from the local training results.

Suppose the SGD method is adopted to train w. The problem can be defined as

follows: each vehicle v_i calculates the gradient or model weight locally with its own data, X_i and Y_i. The central server aggregates all the local gradients or model weights to obtain a temporary global model weight, which is sent back to vehicles for local updating. The processes are carried out iteratively until a convergence condition is satisfied. Then, the training model can be obtained with the final model weight.

In the following, we suppose the vehicles calculate and upload model weights rather than gradients although the proposed framework can support both cases. The local, temporary global, and global model weights are denoted as w_i, w' and w, respectively. A concrete instance is given in Section 3.6.1.

The design goals behind our framework are as follows:

- **Privacy preservation**: a) not only the inputs X_i and Y_i, but also the local model weight w_i from each vehicle v_i should not be revealed to any parties except v_i itself. b) The central server and RSUs cannot infer the identities and the trajectories of vehicles from the communications.

- **Low overhead**: communication and calculation resources are usually limited in vehicular networks. Thus, we aim at minimizing the communication and computation costs of vehicles as much as possible.

- **Reliable communication**: packet loss is a problem in the communications among vehicles. In this work, we aim at providing reliable communications among vehicles for local model weight aggregation. Based on the NEP theory, we focus on the recovery of the critical packets. It is achieved in the application layer as an extra assurance besides the strategies in other lower layers.

3.4.2 Threat model

In this work, there are mainly four entities: RSUs, a cloud server, vehicles and a trusted authority (TA). We formally define and introduce them in Section 3.5.1. In this section, we describe the security assumptions and threat model of the framework.

We assume that the TA is fully trusted. All the other entities except the TA are semi-honest. They may be curious about the local inputs and the local model weights but firmly follow the protocol. The cloud server and the RSUs may also be curious about the identities or trajectories of vehicles. From this point of view, the curious entities can be considered as *internal adversaries* or internal attackers. All vehicles are assumed to be honest on their local data and training results. *External*

adversaries may have the ability to capture messages transmitted between two parties. Based on these assumptions, we target on resisting against the following four major attacks:

- **Data leakage attack**: the raw video recordings may be exposed during the training process.
- **Data inference attack**: both the internal and external adversaries may reconstruct or infer the raw training data from the information obtained. A successful data inference attack results in unintentional data leakage.
- **Identity leakage attack**: the real identities of vehicles may be exposed to RSUs and the cloud server during the training process.
- **Trajectory tracking attack**: the trajectories of vehicles may be tracked by RSUs and the cloud server with the information obtained.

In addition, we take the eavesdropping attack, the message forgery attack, and the message tempering attack into consideration as well. These common attacks focus on the regular communication between parties and are not unique to the scenario of this work.

3.5 The CRS Framework

In this section, we first introduce how to set up the entities in the framework. Then, we present the algorithms to aggregate the local model weights among vehicles securely and reliably. The CRS framework is described in details.

3.5.1 Cluster generation and setup

We first formally introduce the entities and the clusters in CRS as follows:

- **The road-side units**: the RSUs are wireless communicating devices or base stations on the roadside.
- **The cloud server**: the central server CS runs in the cloud computing environment and can communicate with the RSUs efficiently.

- **Vehicles**: vehicles are the workers in the machine learning framework who generate and update local model weights. In this work, a bunch of vehicles traveling in the same direction, $\{v_i \mid i \in N_\mathrm{v}\}$, compose a *cluster*. $N_\mathrm{v} = \{1, 2, \ldots, n_\mathrm{v}\}$ where n_v is the size of the cluster. The *leading vehicle* v_1 is the first one in the traveling direction in the cluster while the *trailing vehicle* v_{n_v} is the last one in the direction. A cluster *head CH* performs as a gateway between the cluster to the nearest RSU. All other vehicles in the cluster are called *cluster members*.

Besides, a **trusted authority (TA)** generates, distributes and manages the cryptographic keys to all valid entities.

We assume the following:

- All the participating vehicles and the server maintain the same training model.
- There are communication strategies working under the application layer to guarantee the stability and reliability of the communication among all the parties as much as possible (such as the MAC-layer retransmission technique in IEEE 802.11).
- We assume the pseudo-IDs of vehicles change in different communication rounds. It can be achieved with any of the many existing methods [111, 67, 77].
- A cluster is stable in a time slot. In other words, we do not consider the member join, the member departure and the head update in this work. As introduced in Section 3.2.2, it should be feasible with the existing techniques. We further discuss the reason in Section 3.8.

Cluster formation and head election have been studied for a long time. It is not the main focus of the work but we need to get the cluster ready for the following training processes. We extract the head election process from the mechanism proposed by Sivaraj *et al.* [101] and present a setup protocol. The cluster setup can be achieved during head election without introducing high overhead. The details of the protocol are as follows:

1. The leading vehicle v_1 chooses a generating element p on a finite cyclic group $\mathbb{G}$ of order q. v_1 chooses a random natural number r_1 and calculates $p^{r_1} \mod q$. A list P is initialized as $\{p^{r_1} \mod q\}$.

2. We denote the weight of vehicle v_i by ρ_i. It reflects the qualification of v_i to be the cluster head based on application demands. For instance, as described in Section 3.3, ρ_i in [101] is calculated with multi-metrics such as the distance. v_1 initializes the current head candidacy $CH = v_1$ and the current head weight $\rho_{CH} = \rho_1$.

3. v_1 broadcasts a packet HELLO. p, q, P and the initialized information of the current head candidacy (i.e., CH and ρ_{CH}) are contained in HELLO.

4. When a vehicle v_i receives HELLO from its one-hop neighbor, it first compares ρ_i with ρ_{CH}. If $\rho_i > \rho_{CH}$, v_i updates $\rho_{CH} = \rho_i$ and $CH = v_i$ in HELLO. Besides, v_i chooses a random natural number r_i and calculates $p^{r_i} \mod q$. v_i appends $p^{r_i} \mod q$ to the list P in HELLO and forwards HELLO to the one-hop neighbors of v_i.

5. Once the trailing vehicle v_{n_v} receives HELLO, it first compares ρ_{n_v} with ρ_{CH}. If $\rho_{n_\mathrm{v}} > \rho_{CH}$, v_{n_v} updates the corresponding fields. Afterwards, CH is successfully selected with the maximum weight.

6. v_{n_v} chooses an integer g (for instance, $g = n_\mathrm{v}$), a random natural number r_{n_v} and a positive integer number R. R should satisfy that $|w_i| \in \mathbb{R}_R^+$ where w_i is any possible model weights. v_{n_v} then appends $p^{r_{n_\mathrm{v}}} \mod q$ to P so that $P = \{p^{r_1} \mod q, p^{r_2} \mod q, \ldots, p^{r_{n_\mathrm{v}}} \mod q\}$. g, P, R, and CH are involved in a packet NOTIFY. v_{n_v} sends NOTIFY to announce the election result among the whole cluster.

7. Every vehicle v_i receiving NOTIFY calculates $p^{r_i r_j} \mod q = (p^{r_j} \mod q)^{r_i} \mod q$ for $\forall j \in N_\mathrm{v} \wedge j \neq i$. We denote $(p^{r_i r_j} \mod q) \mod R$ by α_{ij}.

In the setup protocol, the Diffie–Hellman key exchange protocol is used in a group way to securely exchange $\{p^{r_i r_j} \mod q \mid i \in N_\mathrm{v}, j \in N_\mathrm{v} \wedge i \neq j\}$ among vehicles. Any v_i cannot calculate $p^{r_j r_{j'}} \mod q$ from p^{r_j} and $p^{r_{j'}}$ for any $j \neq i$ and $j' \neq i$. This is guaranteed by the difficulty of solving the **discrete logarithm problem**. With the setup protocol, each vehicle v_i obtains g and $\{\alpha_{ij} \mid j \in N_\mathrm{v} \wedge j \neq i\}$, which are necessary for the following reliable and secure data aggregation algorithms.

3.5.2 Reliable and secure data aggregation

To aggregate all the local model weights $\{w_i \mid i \in N_\text{v}\}$ securely and reliably in a cluster, we propose two algorithms: the message encoding algorithm (i.e., Alg. 3.1) carried out by each vehicle and the data aggregating and recovering algorithm (i.e., Alg. 3.2) carried out by *CH*. In the algorithms, we adopt the data masking technique to preserve the privacy of local model weights. Besides, we provide further reliability of the transmitted model weights in the application layer based on the NEP theory and the network coding technique.

Algorithm 3.1. Message encoding algorithm

v_i masks the local model weights w_i by (3.4) to c_i with $\{\alpha_{ij} \mid j \in N_\text{v} \wedge j \neq i\}$;
v_i sends c_i to *CH*;
v_i calculates $\beta_i = \frac{i-1}{g-1}$;
if v_i is on the routing path from v_j to *CH* and receives c_j **then**
    Calculates $\beta_j = \frac{j-1}{g-1}$;
    Generates an encoded message $\text{COD}(i\|j) = \beta_i c_i + \beta_j c_j$;
    Relays both c_j and $\text{COD}(i\|j)$ to the next hop.
end if
if v_i receives $\text{COD}(l)$ and is on the corresponding routing path where l is a list of vehicle indexes **then**
    Generates an encoded message $\text{COD}(i\|l) = \beta_i c_i + \text{COD}(l)$;
    Relays both $\text{COD}(l)$ and $\text{COD}(i\|l)$ to the next hop.
end if

Algorithm 3.2. Data aggregating and recovering algorithm

CH tries to collect all $\{c_i \mid i \in N_\text{v}\}$ from vehicles in a given time threshold Δt;
CH drops the unnecessary $\text{COD}(l)$ (i.e., all $\{c_i \mid i \in l\}$ have been received) when it arrives;
for each c_i which is not received in Δt **do**
    CH tries to recover c_i by solving one or multiple linear equations $\text{COD}(l)$ that satisfy $i \in l$;
end for
if $\forall c_i$ are received or $\forall c_i$ that can be recovered are recovered **then**
    CH calculates the average of all w_i ($i \in N_\text{v}$), denoted as w_local, by (3.5).
end if

In Alg. 3.1, each vehicle v_i first masks w_i based on (2.1) as follows:

$$c_i = w_i + \sum_{j \in N_\text{v}: i<j} \alpha_{i,j} - \sum_{j \in N_\text{v}: i>j} \alpha_{j,i} \pmod R \tag{3.4}$$

R is a control parameter which limits the size of the masked value. Note that $\alpha_{i,j} = \alpha_{j,i}$ in our work. Because the indexes or IDs of vehicles and g are public to every vehicles, v_i can calculate β_i and necessary β_j as Steps 3 and 5. β_i and β_j are used to encode raw messages based on network coding, i.e., to generate linear equations. As a routing node, v_i is responsible to relay both the original messages and the encoded messages, as described in Steps 4–11.

The encoded messages are used to recover the lost messages in the application layer, as shown in Steps 3–5 in Alg. 3.2. This is achieved by solving the linear equations COD(l) where l is a list of vehicle indexes. The unknowns of the equations are the lost c_i and the coefficients are $\{\beta_j \mid j \in l\}$. To save the memory space, *CH* can drop the unnecessary COD(l) directly as described in Step 2. Once all c_i are received or recovered, *CH* calculates the average of all w_i based on (2.2) as follows:

$$w_{\text{local}} = \frac{\sum_{i \in N_{\text{v}}} w_i}{n_{\text{v}}} = \frac{\sum_{i \in N_{\text{v}}} c_i}{n_{\text{v}}} \tag{3.5}$$

where the randomness is canceled but the real model weights of any member vehicles are not exposed to *CH*.

3.5.3 Two-layered secure learning framework

The presented two-layered secure learning framework, CRS, has a "vehicle clusters-RSUs-cloud server" architecture, as shown in Fig. 3.1. The first layer is composed of vehicle clusters while the second layer is RSUs and the server *CS*.

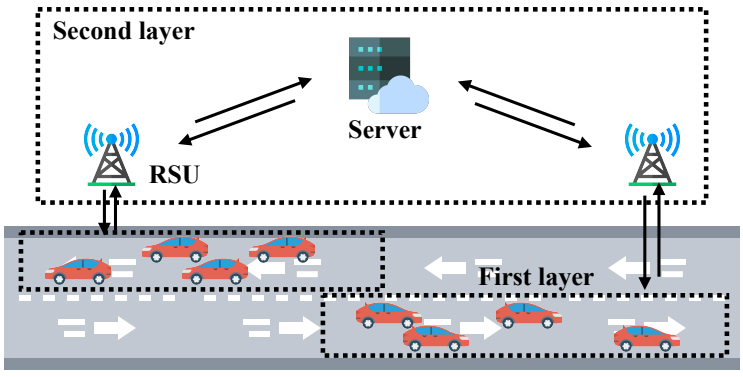


Figure 3.1: Architecture of CRS

The framework is described in Fig. 3.2. In the first layer, as shown in Steps 1–4, the local model weights are masked and aggregated by *CH* securely. In Steps 5–6,

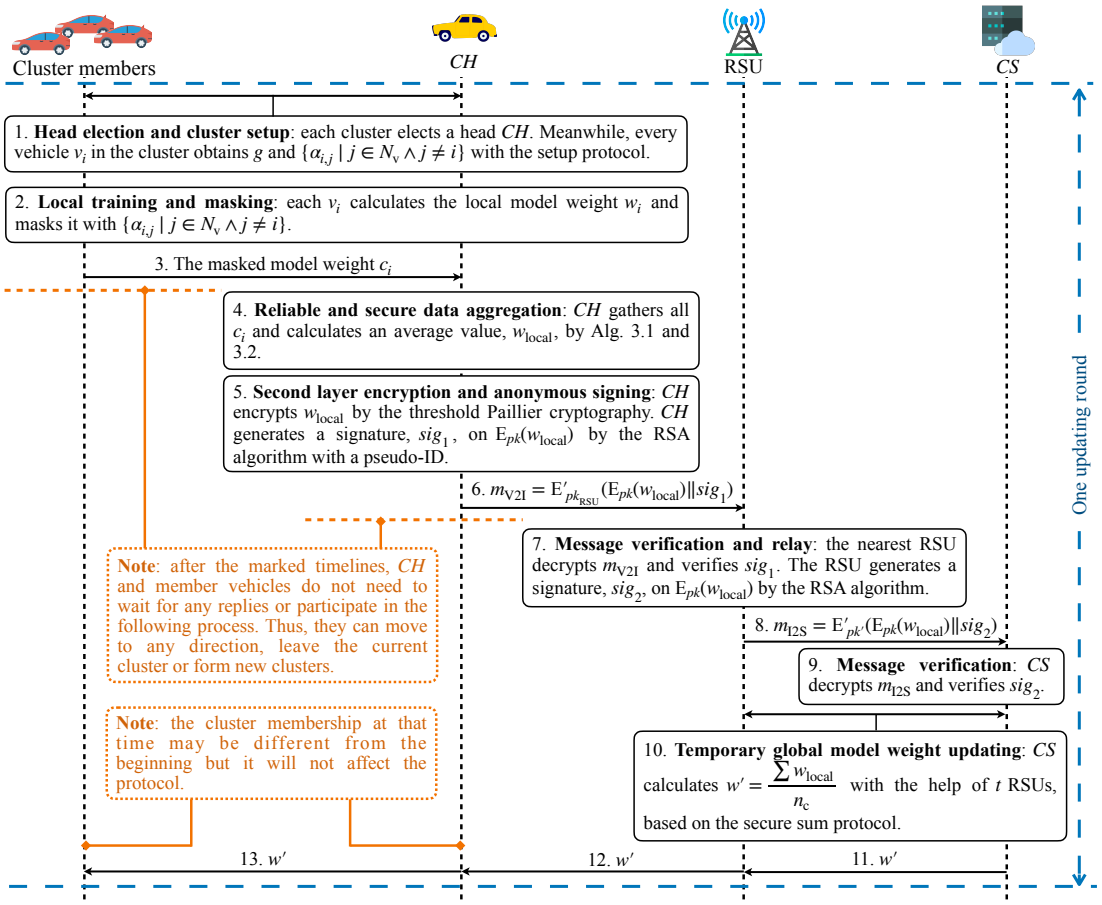


Figure 3.2: CRS Framework

after getting the local average value w_{local}, *CH* encrypts it by the threshold Paillier cryptosystem. We denote the encryption result as $\text{E}_{pk}(w_{\text{local}})$ where $pk = (pk_1, pk_2)$ is the public key pair of *CS*. Note that pk is public to all entities in CRS while the corresponding secret key shares are only maintained by the RSUs. The key generation and distribution are completed by the TA in advance. We omit the descriptions for simplicity. To preserve the identity and trajectory privacy, *CH* generates an anonymous communication channel with the nearest RSU. To be specific, *CH* signs $\text{E}_{pk}(w_{\text{local}})$ with a pseudo-ID. The anonymous signature is denoted by sig_1. To resist against the man-in-the-middle attack, we adopt the sign-then-encrypt mechanism. Both the message $\text{E}_{pk}(w_{\text{local}})$ and sig_1 should be encrypted by the ElGamal algorithm as $m_{\text{V2I}} = \text{E}'_{pk_{\text{RSU}}}(\text{E}_{pk}(w_{\text{local}}) \| sig_1)$ where $\|$ means concatenation. $\text{E}'_{pk_{\text{RSU}}}$ is the ElGamal encryption function with RSU's public key pk_{RSU}.

To relay the local weight to the server securely, in Steps 7 and 8, the RSU gener-

ates a signature, sig_2, on $\mathrm{E}_{pk}(w_{\mathrm{local}})$. Similarly, both $\mathrm{E}_{pk}(w_{\mathrm{local}})$ and sig_2 should be encrypted with *CS*'s public key pk' by the ElGamal algorithm. Please note that we use pk and pk' to distinguish the public keys of *CS* used in the threshold Paillier encryption and the ElGamal encryption, respectively. In the second layer, *CS* chooses t RSUs to calculate $\sum w_{\mathrm{local}}$ without knowing the plaintext of any w_{local} (i.e., Step 10), as described in Section 3.3. t is the threshold we adopted. The temporary global model weight can be updated as $w' = \frac{\sum w_{\mathrm{local}}}{n_{\mathrm{c}}}$ where n_{c} is the number of clusters. Then, RSUs relay w' to all the current clusters for local model weight updating (i.e., Steps 11–13). The processes are repeated (we refer to it as one *updating round*) until a convergence condition is satisfied.

With the two-layered framework, the local model weight of a vehicle is not revealed to any parties except the vehicle itself. The average model weight of a cluster is not revealed to both RSUs and *CS*. The member vehicles are totally hidden behind *CH* and *CH* itself is anonymous to RSUs and *CS*. Thus, the privacy of vehicle identity and trajectory is preserved.

3.6 Application and Prospects

3.6.1 An application instance

In this section, we consider a simple application to detect traffic accidents from dashcam videos as an instance.

We formally define X_i and Y_j, which have been introduced in Section 3.4.1, as follows: $X_i = \{X_{i,1}, X_{i,2}, \ldots, X_{i,n_{\mathrm{r}}}\}$ denotes the original video recording set of v_i. The total number of videos v_i keeps is n_{r}. $N_{\mathrm{r}} = \{1, 2, \ldots, n_{\mathrm{r}}\}$. The information extracted from a video $X_{i,j}$ that $j \in N_{\mathrm{r}}$ is denoted as $x_{i,j}$. $Y_i = \{y_{i,1}, y_{i,2}, \ldots, y_{i,n_{\mathrm{r}}}\}$ represents the corresponding label set of X_i. Any $y_{i,j}$ ($j \in N_{\mathrm{r}}$) satisfies $y_{i,j} \in \{1, 0\}$ where 1 represents *traffic accident* while 0 represents *normal driving*. Thus, each pair of $(x_{i,j}, y_{i,j})$ can be treated as a training sample. The formal machine learning goal is to train a model classifying a dashcam video into the categories of traffic accident or normal driving.

Before introducing the training model, we first describe the pre-processing of videos. To generate $x_{i,j}$ from a video $X_{i,j}$, we extract frames from the video, convert the frames to gray scale and resize each of them to the resolution of 256×144. These are critical pre-processing steps in computer vision which can reduce the data

amount and save the training time. With the pre-processing, $x_{i,j}$ can be represented as $\{fra_1, fra_2, \ldots, fra_{n_\mathrm{f}}\}$ where fra_i is the i-th frame and n_f is the number of frames in $X_{i,j}$. Each of fra_i is composed of 256×144 pixels in the range of $[0, 255]$.

Many researchers work on designing a machine learning model for traffic accident detection [48, 35, 98, 25, 47, 104]. To achieve higher classification accuracy, these works may extract different features, involve different techniques, or adopt different neural network layers. In this instance, we adopt a hierarchical recurrent neural network (HRNN)-based model [35]. The architecture of the model is not complicated but the classification performance is good.

The model adopts the long short-term memory (LSTM). LSTM is a type of recurrent neural network capable of learning long-term dependencies and suitable for video handling. To be specific, the training data, $(x_{i,j}, y_{i,j})$ for all videos, are input to the first time-distributed LSTM layer with 64 hidden states, which analyzes the time-dependent sequence of the frames. The extracted temporal features between each frame are passed through to the second LSTM layer with 64 hidden states, where features for classification are learned. The last layer is a dense layer with softmax activation for classifying videos. In this deep learning model, the model weight w_i is the vector containing the trainable parameters of each layer. For each vehicle v_i, the training is an iterative process where w_i is updated to minimize a local empirical risk:

$$\mathrm{F}_i(w_i) = \frac{1}{n_\mathrm{r}} \sum_{j \in n_\mathrm{r}} \mathrm{f}_{i,j}(w_i) \tag{3.6}$$

$\mathrm{f}_{i,j}(w_i)$ is the cross-entropy loss function. It reflects the difference between the classification result and the real label of the data $x_{i,j}$. Thus, to find w_i that minimizes $\mathrm{F}_i(w_i)$ equals to find w_i that has a better classification accuracy. For brevity, we omit the detailed introduction of neural networks and the related concepts in this section.

To solve the optimization problem, we adopt the SGD and the idea of the FedAvg model [71] for the local update. FedAvg is a popular and leading method because it reduces the global convergence time and the number of communication rounds. In our work, each vehicle locally updates w_i by Alg. 3.3. The main idea is to update w_i multiple times with random data batches before sending it to CH.

After the local updating, the process is exactly the same as that described in Section 3.5.3. We do not repeat the CRS workflow but summarize the whole procedure of the application as follows:

Algorithm 3.3. Vehicle local update algorithm

Input: Current updating round t_u, local training samples $\{(x_{i,j}, y_{i,j}) \mid j \in \{1, 2, \ldots, n_{\mathrm{v}}\}\}$, number of batches n_{B}, number of local epochs n_{E}

Output: Updated local model weight vector w_i

v_i shuffles the local training samples;
if $t_u = 0$ **then**
    Pick a random w_i;
else
    Obtain the temporary global weight vector w' from the nearest RSU as the current w_i: $w_i \leftarrow w'$;
end if
Split the local data set to n_{B} batches: $\{B_1, B_2, \ldots, B_{n_{\mathrm{B}}}\} \leftarrow \{(x_{i,j}, y_{i,j}) \mid j \in \{1, 2, \ldots, n_{\mathrm{v}}\}\}$;
for each local epoch $E \in \{1, 2, \ldots, n_{\mathrm{E}}\}$ **do**
    for each batch $B \in \{B_1, B_2, \ldots, B_{n_{\mathrm{B}}}\}$ **do**
        Update w_i with the HRNN-based model with B;
    end for
end for
return w_i

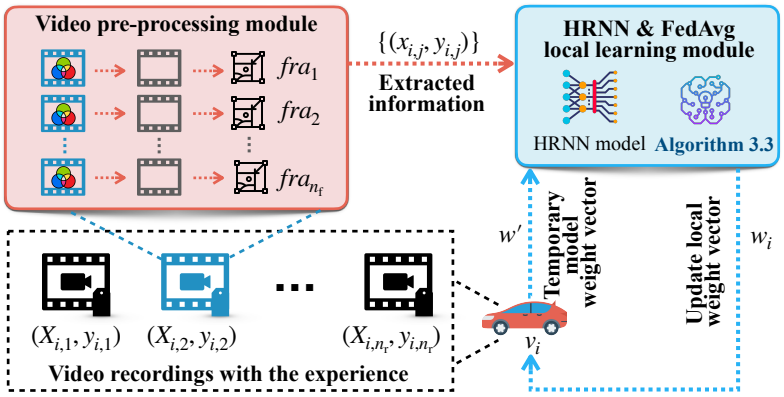
1. As shown in Fig. 3.3a, vehicles first handle the local video recordings with the video pre-processing module, where $x_{i,j}$ is generated by frame extracting, gray scale converting and frame resizing.

2. The output information $x_{i,j}$ with the corresponding label $y_{i,j}$ is used as the training samples of the HRNN-based model. Each vehicle locally updates the local weight vector w_i with Alg. 3.3. The task is executed by the HRNN and FedAvg local learning module with a temporary global model weight vector w'.

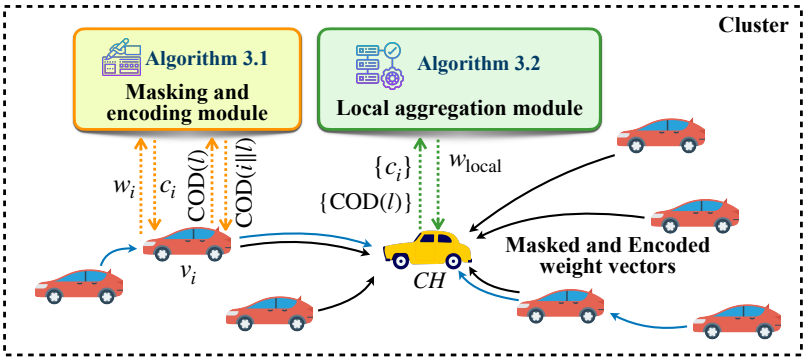
3. The updated w_i is first masked by the masking and encoding module, which preserves the privacy of local data and training results. Suppose there are n_{p} parameters in the HRNN model. The masked value c_i is a column vector with n_{p} entries. As shown in Fig. 3.3b, each vehicle sends c_i to *CH*. To provide further guarantee of reliable communication, the member vehicles are also responsible for encoding and relaying the passing messages COD(l) as explained in Alg. 3.1.

4. *CH* gathers and recovers all the local c_i with Alg. 3.2. The averaged local weight vector w_{local} is calculated by the local aggregation module.

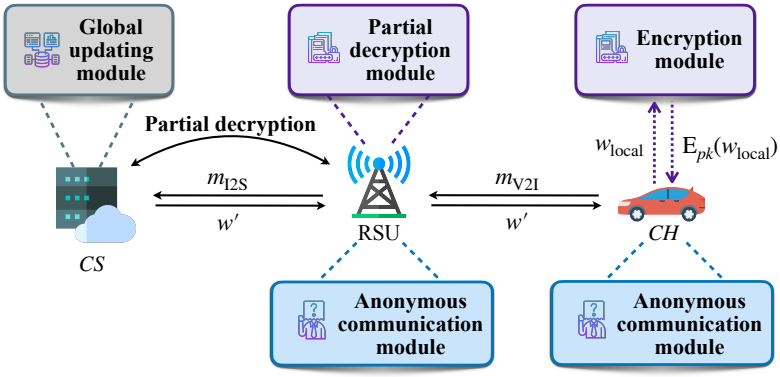
5. As shown in Fig. 3.3c, w_{local} is first encrypted by the threshold Paillier cryp-



(a) Video pre-processing and local training



(b) Cluster aggregation



(c) Global updating

Figure 3.3: CRS application example

tosystem in the encryption module. Afterward, *CH* sends m_{V2I}, which contains the cipertext of w_{local}, to the nearest RSU by the anonymous communication module.

6. *CS* gathers all the ciphertexts and handles them in the global updating module.

The decryption and averaging process needs the help of RSUs, who have the ability of partial decryption. CS informs the calculated temporary global weight vector w' to all participating vehicles, which leads to a new updating round.

3.6.2 More use cases and prospects

We have described an application instance of traffic accident detection with details. Besides that, the proposed framework can also be adopted into many other intelligent transportation applications, with the informative videos and other valuable data provided by vehicles. In this section, we further discuss the use cases and the development prospects in IoV, especially in autonomous driving.

In autonomous driving, a critical task is the wheel steering angle prediction. With the prediction, proper actions can be taken to control the steer and keep the vehicle in the lane. With machine learning, we can train a model that takes the past and the current information to predict the angle and make decisions. This task requires not only environmental data (such as the images taken by on-board cameras) but also driving data (such as steering wheel angle, throttle and speed). The real decisions made by drivers when manually driving are used as ground truth. It has been studied in many existing works [127, 128].

Definitely, training such a model with good performance requires the participation of a huge number of vehicles and drivers. However, privacy leakage and communication cost are major concerns if all the local data are transmitted to a central server. Adopting the proposed two-layered distributed framework, drivers do not need to upload their local data. They can drive vehicles as usual while the machine learning model is securely trained for autonomous driving. In addition, the distributed manner allows vehicles to always keep the up-to-date model, which is critical for robustness and safety purposes.

Besides, with the rich data provided by vehicles, the proposed framework can be adopted for other purposes as well, including obstacle avoidance, route planning, behavior selection, intersection handling and so on. In all of these use cases, the framework allows vehicles to train analytical models in a privacy-preserving way and use the latest models right away.

In industry, applying such an IoV framework is significantly valuable. For example, Motional begins conducting autonomous deliveries for Uber Eats customers in 2022 [78]. We can see a promising future if they involve both the regular deliv-

ery vehicles and the autonomous vehicles for distributed learning, and probably even encourage their customers to participate in the training process. The two-layered framework does not expose the privacy of drivers but makes the best use of the local data to improve the performance of self-driving. Besides, other companies such as Ford Motor, Volvo Cars, Mercedes-Benz, BMW and Google have projects on autonomous driving as well. In the future, the huge amounts of driving data with distributed learning will further promote the popularization of self-driving cars and the development of the intelligent transportation system.

3.7 Performance Evaluation

In this section, we first introduce the simulation setup in ns-3.34. The simulations are conducted in two scenarios: a) an **emulated scenario**, where we can have full control over the vehicles and observe the impact of vehicle speed clearly, and b) a **realistic scenario** in Bologna, Italy, which is more complex and can evaluate the practicability of CRS in the real world. The packet recovery performance, and the communication and computation costs are evaluated. Besides, we analyze the privacy achievements of CRS, followed by a performance evaluation summary and comparison. Note that all results are calculated or simulated for one updating round.

3.7.1 Simulation setup

Emulated highway scenario

Considering the scenario where vehicles travel in the same direction along a straight highway, the total number of vehicles in a cluster is set to 20. As shown in Fig. 3.4, there are two traffic lanes on a highway and the width of each lane is 3.7m (based on the Geometric Design Guide for Canadian Roads [108]). All vehicles travel at a specific speed which may be different from that of others. The speed follows a truncated Normal distribution $\mathcal{TN}(80, std, 60, 100)$ where 80km/h is the mean value, std is the standard deviation, 60km/h is the lower bound and 100km/h is the upper bound. In Sections 3.7.2, 3.7.3 and 3.7.4, we set different std to observe the influence of vehicle speed on the performance of CRS. According to the two-second rule, the initial trailing distance between every two vehicles is around 44m (i.e., 2s with 80km/h). Thus, the length of the cluster is around 400m. The standard of IEEE 802.11p is adopted for vehicle-to-vehicle (V2V) communication and the communication range

is set to 100m. Because the cluster head election process depends on the particular strategies chosen, we set a vehicle located around the center of the cluster (i.e., the yellow vehicle in Fig. 3.4) as *CH* for simplicity.

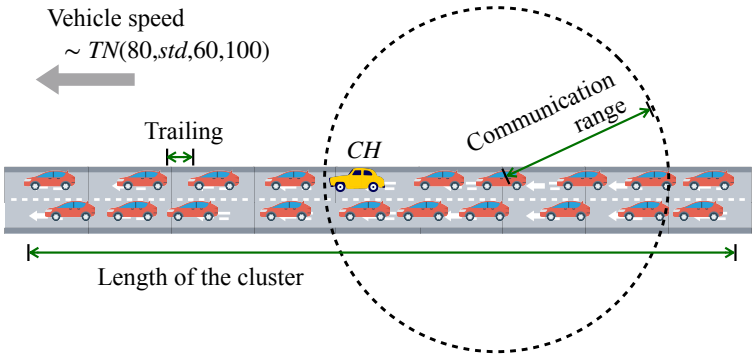


Figure 3.4: Vehicle position and mobility in the emulated scenario

In the second layer, the vehicle-to-infrastructure (V2I) communication adopts IEEE 802.11p as well, with a communication range of 500m. We consider the distance between two RSUs as 800m, i.e., the distance between *CH* and the nearest RSU is in the range of 0m to 400m. We consider two scenarios where the server collects data from a city (e.g., Bologna) or from a country (e.g., Italy), separately. Thus, the distance between the cloud server and an RSU is set as half of the furthest straight-line distance of Bologna (i.e., 10.59km) and that of Italy (i.e., 592.50km), respectively. The propagation speed equals the speed of light, i.e., 299792.46km/s.

The size of the weight vector w_i varies with different applications and learning algorithms. For example, suppose 48 frames can be extracted from each video recording. With adopting the deep learning model illustrated in Section 3.6.1, the length of w_i is 115330. We refer to it as a *large* w_i in the following. In this section, we first consider w_i as a vector with only one entry as a representative for evaluation. Then, we briefly give the main simulation results with the large w_i adopted for reference in Section 3.7.5.

Realistic city scenario

Vehicle mobility has a considerable influence on intra-cluster communication. In the real world, the mobility and trajectories are usually more complex than the settings in the emulated scenario. Vehicles traveling in busy parts of a city may also take more acceleration and deceleration than the vehicles on a highway. Thus, to further evaluate

the performance of CRS, especially the intra-cluster communication performance, we introduce an urban city scenario with real-world traffic networks, mobility data and trajectories [12]. The realistic scenario simulates one-hour traffic in the city of Bologna, Italy. The real-world datasets used are supplied by the municipality of Bologna. We keep the settings the same with that in the emulated scenario for result comparison. Thus, we still select clusters with 20 vehicles. The V2V communication range is 100m. To simulate the V2I communication, we manually set an RSU in the traffic network with the same settings in the emulated scenario.

Note that, in the realistic scenario, we mainly focus on the packet recovery performance and the intra-cluster communication cost. The communication in the second layer, i.e., the communication between RSUs and *CS*, is simulated as described in Section 3.7.1. The reasons are twofold: a) most of the public realistic traffic datasets only provide real-world data of vehicles but do not provide data of RSUs and servers; b) in this work, packet loss is only considered within a cluster. The mobility of vehicles has obvious impact on the intra-cluster communication but no impact on the second-layer communication.

All simulations in both the emulated and the realistic scenarios are conducted 50 times (i.e., 50 independent *repetitions*) to get an average result.

Cryptographic operations with MIRACL [76]

To evaluate the computation cost of CRS and compare it with other works [84, 90], we simulate the execution time of the cryptographic operations with MIRACL. Results are listed in Table 3.2. We choose 4096-bit pk_2 to provide 128-bit security in the Paillier cryptosystem. The hardware used for the simulation features an Intel Q9550 CPU with 2.83 GHz operating frequency, and 8GB RAM.

3.7.2 Packet loss and recovery

The encoded messages can help to recover the lost c_i. To evaluate the performance, we observe the results with different packet loss rate. In the simulation, many possible reasons may lead to packet loss, such as collisions, invalid routing and congestion. To control the *actual packet loss rate*, we drop additional packets with a *constant drop rate* in the IP layer.

In the emulated scenario, the constant drop rate is set to 1% and 2%. In addition, we adopt different *std* to simulate different mobility models. As shown in Fig. 3.5a,

Table 3.2: Cryptographic operations and execution time [76] in Chapter 3

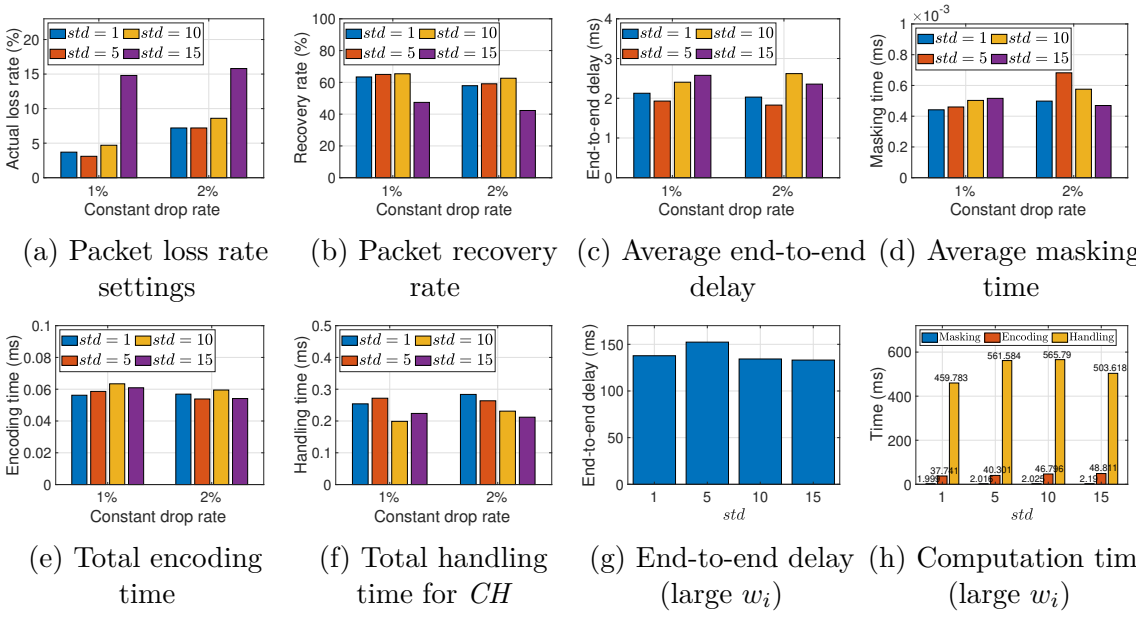
Operation	Abbreviation	Time (ms)
Scalar addition on $\mathbb{Z}_R$	$T_{\text{Z\_ADD}}$	0.0037
Scalar multiplication on $\mathbb{Z}_R$	$T_{\text{Z\_MUL}}$	0.0046
Scalar multiplication on $\mathbb{Z}_{pk_2}$	$T_{\text{pn\_MUL}}$	1.5832
Scalar exponentiation on $\mathbb{Z}_{pk_2}$	$T_{\text{pn\_EXP}}$	20.5559
Scalar multiplication on $\mathbb{Z}_{pk_2^2}$	$T_{\text{pn2\_MUL}}$	5.9248
Scalar exponentiation on $\mathbb{Z}_{pk_2^2}$	$T_{\text{pn2\_EXP}}$	71.5902
ElGamal-2048 encryption	$T_{\text{EG\_E}}$	45.6259
RSA-2048 encryption	$T_{\text{RSA\_E}}$	2.8275
RSA-2048 decryption	$T_{\text{RSA\_D}}$	568.1378
Matrix multiplication †	$T_{\text{l\_mMUL}}$	2668.6263
Scalar multiplication †	$T_{\text{l\_sMUL}}$	0.8122
Matrix addition †	$T_{\text{l\_ADD}}$	0.8391
Gaussian elimination	T_{GE}	0.0117

† *Related to the lattice-based cryptography. Parameters are chosen to provide 128-bit security [84].*

the actual loss rate is around 3% to 10% when *std* is 1, 5 and 10, which is reasonable in VANETs. When $std = 15$, the actual loss rate goes as large as 15%, which is an extreme scenario. The packet recovery rate is given in Fig. 3.5b. In most of the cases, the average recovery rate is around 60%. It is a little lower when $std = 15$ because the routing topology is changing more frequently and leads to a more complex scenario. The results show that the mobility model has an impact on packet loss rate and recovery rate, which echoes our motivation of introducing a real-world datasets-based scenario.

In the realistic scenario, we further set the constant drop rate to 0%, 1%, 2% and 3% so that the actual loss rate varies from 2% to 10%. As shown in Fig. 3.6a, the average recovery rate is around 60% with the loss rate from 4% to 10%, which echoes the results in the emulated scenario with $std = 1, 5$ and 10. When the constant drop rate is 0%, the actual loss rate is as less as 2%, i.e., usually only one or zero w_i is not received by CH. The low loss rate leads to a relatively lower recovery rate (around 40%). It is because the recovery relies on the existing of routes, whose probability is lower when there is only one packet loss.

The packet recovery performance is acceptable in this work. The reasons are

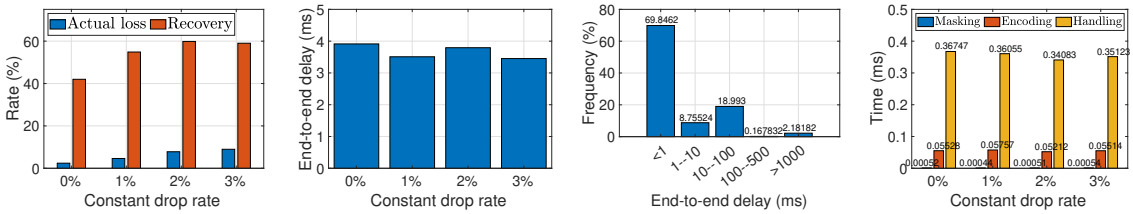


(a) Packet loss rate settings (b) Packet recovery rate (c) Average end-to-end delay (d) Average masking time

(e) Total encoding time (f) Total handling time for CH (g) End-to-end delay (large w_i) (h) Computation time (large w_i)

Figure 3.5: Simulation results of CRS with the emulated highway scenario

twofold: a) we only aim at providing an additional guarantee of the communication in the application layer and reducing the loss of the most valuable packets as much as possible. b) The training process in machine learning requires multiple updating rounds. The loss of a very few local weight vectors makes a negligible difference on the final training result from a long-term perspective. Additionally, we discuss the possible improvements in Section 3.8.



(a) Packet loss and recovery (b) Average end-to-end delay (c) End-to-end delay histogram (d) Computation time

Figure 3.6: Simulation results of CRS with the realistic scenario

3.7.3 Communication cost

With the cluster setup protocol, necessary parameters are broadcast. For the sake of security, we choose a 2048-bit q [1] so that the length of each p^{r_i} is 2048 bits

and that of P is at most $2048n_{\mathrm{v}}$ bits. Thus, the total size of a packet HELLO or NOTIFY is $\mathcal{O}(n_{\mathrm{v}})$ bits. Taking the emulated scenario with $std = 1$ and the realistic scenario as representatives, the end-to-end delay between two vehicles is 11.48ms and 10.16ms, respectively, on average. The total communication time (from the leading vehicle sending out the first HELLO packet until the trailing vehicle receives all the packets) is 306.36ms and 566.12ms, respectively. The difference is mainly caused by the different trailing distance and the number of one-hop neighbors in the two scenarios. The results show that the setup does not introduce high cost to the head election process.

In the first layer, besides c_i, each member vehicle may also need to generate and send encoded messages as a router. The total number of messages sent in the cluster is $\mathcal{O}(n)$. Although the message encoding algorithm leads to more communication cost, it is still acceptable. We discuss more about the trade-off between the recovery rate and the cost in Section 3.8.

In the cluster, the average end-to-end delay for every vehicle in the emulated scenario and the realistic scenario is usually less than 4ms, as shown in Figs. 3.5c and 3.6b. Note that, in the simulations, there are a few irregular results: some vehicles may have an obvious long end-to-end delay in some repetitions, even more than 1000ms. These unusual results are caused by the failure of address resolution or the MAC-layer retransmission. Taking the realistic scenario with 0% constant drop rate as an example, the frequency histogram of all the end-to-end delay records is given in Fig. 3.6c. It can be seen that there are only 2% delay records larger than 100ms approximately. Thus, all the average end-to-end delay is calculated only with the common and typical results (i.e., less than 100ms). The results show that the communication cost for vehicles is relatively low in both scenarios, and the mobility model and the actual loss rate do not have an obvious effect on it.

The ciphertext $\mathrm{E}_{pk}(w_{\mathrm{local}})$ is generated by *CH* with the threshold Paillier cryptosystem. Suppose a 4096-bit pk_2 is chosen, each ciphertext is 8192 bits. The size of the anonymous signature sig_1 is 2048 bits with RSA-2048. $\mathrm{E}_{pk}(w_{\mathrm{local}})$ and sig_1 are encrypted by the ElGamal scheme, which doubles the size. Thus, we use 20480-bit (i.e., $2 \times (8192 + 2048)$) messages as an example to evaluate the communication time between *CH* and the nearest RSU. The average end-to-end delay is 7.46ms and 7.48ms in the emulated and realistic scenarios, respectively.

In the second layer, similarly, the message sent from an RSU to the server is 20480 bits. Besides, t RSUs need to participate in the decryption of $\mathrm{E}_{pk}(w_{\mathrm{local}})$. The

number of communication rounds for each RSU is $\mathcal{O}(1)$. TCP (Transmission Control Protocol) is adopted in the RSU-to-server simulation. The end-to-end delay is around 0.41ms and 15.08ms for citywide and countrywide data collection, respectively.

3.7.4 Computation cost

In the cluster setup protocol, we set up the vehicles when electing a head. The head election cost depends on specific strategies, which is not the main focus of our work. Thus, we care more about the additional cost introduced by the setup, specifically, the Diffie-Hellman key exchange protocol. According to the protocol, each vehicle v_i requires n_{v} modular exponentiations on $\mathbb{G}_q$. The total calculation time for each vehicle is around 44.62ms in the simulation.

For every vehicle, n_{v} additions on $\mathbb{Z}_R$ are required to mask w_i. The computation cost to generate the encoded messages $COD(l)$ depends on the number of messages relayed by a vehicle. In our simulation, the number of hops in a route from a member vehicle to *CH* is usually less than 3. Thus, we use the computation cost of the last router vehicle in the route as a representative: 5 multiplications and 3 additions on $\mathbb{Z}_R$ are required. Other vehicles are expected to have less computation cost than it.

In our simulation, Gaussian elimination is adopted to solve the linear equations by *CH*, which takes around 0.0117ms. Besides, there are $n_{\mathrm{v}} - 1$ additions for *CH* to unmask and get the sum of all w_i.

With the emulated scenario, the average masking time for a vehicle, the total encoding time for a vehicle, and the total handling time for *CH* are shown in Figs. 3.5d, 3.5e and 3.5f. Fig. 3.6d shows the corresponding results with the realistic scenario. It can be observed that, in both scenarios, a vehicle can mask w_i in 0.0007ms and encode all passing packets in 0.06ms approximately. The handling time of *CH* is less than 0.4ms, including handling the received encoded messages, preparing the linear equations and solving the equations. The mobility model and the packet loss rate have no obvious effect on these results.

With the Paillier cryptosystem, the encryption of w_{local} requires 1 exponentiation and 1 multiplication on $\mathbb{Z}_{pk_2^2}$, and 1 exponentiation on $\mathbb{Z}_{pk_2}$. Using the Paillier Threshold Encryption Toolbox [107], the encryption time for *CH* is around 255.52ms. The partial decryption time for an RSU is around 1846.89ms. The server *CS* is required to combine all the partial decryptions from the t RSUs. Considering *CS* is configured on cloud with sufficient computation resources, this task is not a bottleneck and can

be achieved efficiently.

3.7.5 Evaluation with the large w_i

As we mentioned in Section 3.7.1, the w_i is a vector with 115330 entries for the application instance. In practice, model compression techniques should be adopted to reduce the size of the deep learning model [50]. For example, model pruning drops out the parameters which are less important while maintaining a similar accuracy as the original model [38]. The existing pruning techniques for distributed machine learning [49, 96, 51] are mature enough to be adopted in our framework. Thus, we treat w_i as a vector with 10% sparsity. In other words, we consider that the vector has $115330 \times 10\% = 11533$ entries after pruning. Because the compression techniques are beyond the scope of our work, we do not give details here but discuss more in Section 3.8. Considering that we have evaluated our work with the small w_i as a representative in details, we only give the primary communication and computation time with the large w_i in the emulated scenario as a reference. The simulations are conducted with a constant drop rate of 0.01.

In the first layer of CRS, as shown in Fig. 3.5g, the average end-to-end delay of a member vehicle is less than 150ms in the simulation. The ciphertext of w_{local} is 11533×8192 bits with a 4096-bit pk_2. With a 2048-bit signature, the size of the message is around 11.81 MB. The ElGamal encryption doubles the size so that the encrypted message sent from *CH* to the RSU is around 23.62 MB. The average time of sending the message is 66.89s. In the second layer, it takes an RSU 74.74s and 272.25s on average to send the message to a server located in Bologna and Italy, respectively. The time is still shorter than directly collecting video recordings.

As shown in Fig. 3.5h, the masking time and the encoding time are less than 3ms and 49ms, respectively. The handling time for *CH* increases obviously compared with that shown in Fig. 3.5f. The main reason is that transforming the received larger messages to linear equations takes a longer time.

3.7.6 Security and privacy analysis

The sign-then-encrypt mechanism protects the entities against the eavesdropping attack, the message forgery attack, and the message tempering attack in communication. Considering that it is a common and typical mechanism adopted in practice, we do not further analyze the security in details. In this section, we mainly describe the

achievement of the privacy-preservation goals defined in Section 3.4.1 and the defense against the major four attacks listed in Section 3.4.2.

Privacy of local inputs and weight vectors

In CRS, the privacy of local weight vectors $\{w_i \mid i \in N_\mathrm{v}\}$ is guaranteed by the masking algorithm. The security of masks $\{\alpha_{i,j} \mid i \in N, j \in N_\mathrm{v}, i \neq j\}$ is twofold: a) the masks are only used for one communication and never reused in a cluster. It works as a one-time pad. b) They are generated via the Diffie-Hellman key exchange protocol. The security of the protocol is based on the difficulty of solving the Discrete Logarithm problem. It has been proved that the protocol is considered secure against eavesdropping when 2048-bit long parameters are chosen [1].

It is infeasible for any parties except v_i to obtain all the masks $\{\alpha_{i,j} \mid j \in N_\mathrm{v}, i \neq j\}$ or the secret value (i.e., r_i) used to generate the masks. With the masks, v_i generates and sends c_i (i.e., the masked w_i) to *CH*. It is infeasible for any other parties to remove the randomness and infer w_i from c_i. Besides, it is unnecessary for *CH* to acquire any w_i or $\alpha_{i,j}$. *CH* can calculate the average of all $\{w_i \mid i \in N_\mathrm{v}\}$ only with $\{c_i \mid i \in N_\mathrm{v}\}$ as described in Alg. 3.2.

Overall, with CRS, the raw local data are never shared with others. The local weight vectors are protected. The first privacy-preservation goal is achieved. The **data leakage attack** and **data inference attack** are defended against.

Privacy of vehicle identities and trajectories

In CRS, all the member vehicles are hidden behind *CH* so that both *CS* and RSUs do not know the exact provider of the local weight vectors. Besides, *CH* communicates with RSUs through an anonymous channel so that the real identity of *CH* is also preserved. Overall, CRS protects all vehicles from the **identity leakage attack**.

An adversary can only infer the trajectories of a vehicle when the vehicle appears in different locations. The member vehicles are never exposed to the RSUs or *CS* with the two-layered architecture so that the privacy of the trajectories is preserved. The trajectories of *CH* are protected by the anonymous communication technique. Recall that we assume the pseudo-IDs of *CH* change in different communication rounds, which provides the properties of unlinkability. The RSUs cannot extract the real identity of *CH* or link two communications from it. Thus, the **trajectory tracking attack** is defended against and the second privacy-preservation goal is achieved.

3.7.7 Summary and comparison

Two privacy-preserving machine learning frameworks are adopted for comparison. One framework (we name it **PPDL**) [84] is a typical framework with cryptographic solutions and another, **SP-CIDS** [90], is specifically designed for VANETs. Both of the works are briefly introduced in Section 3.2.1.

Table 3.3: Comparison on security, privacy and accuracy

	Framework	PPDL [84]	SP-CIDS [90]	CRS
Security and privacy goals	No local weight vector leakage to server	✓ ‡	NA §	✓
	No local weight vector leakage to vehicles	✗ ¶	✓	✓
	Identity preserved	✗	✗	✓
	Location preserved	✗	✗	✓
Accuracy	No packet loss	✓	✗	✓
	Packet loss	✗	✗	✓ ‖

‡ *The goal or property is achieved or can be guaranteed.* § *Not applicable.*
¶ *The goal or property is not achieved or cannot be guaranteed.* ‖ *A solution to handle the packet loss has been proposed.*

As shown in Table 3.3, CRS achieves more security and privacy goals. In addition, as analyzed in Section 3.2.1, providing sufficient and accurate local model weights is important for model training in DML. When there is no packet loss, both PPDL and CRS can guarantee the accuracy of the aggregated model weight w' in the server [3]. However, SP-CIDS adopts the differential privacy technique so that the accuracy of w' is reduced. When there exists packet loss (i.e., part of the local model weights w_i are lost during transmission), all of PPDL, SP-CIDS and CRS cannot guarantee the accuracy of all w' and may require more updating rounds to merge [4]. However, CRS handles the packet loss in the application layer to provide accurate w' as much as possible.

The critical communication and computation costs are summarized in Tables 3.4 and 3.5, respectively. The number of entries of the vector is considered as one as a representative for comparison. All the costs of PPDL, SP-CIDS and CRS are

[3]We use the term *accurate* to describe that w' equals the average of ALL w_i without any perturbation or loss, i.e., $w' = \frac{1}{n_c}\sum_{N_c}\frac{1}{n_v}\sum_{N_v} w_i$.

[4]The authors [90] mention that the missed training data can be replaced with some method such as the mean value substitution. However, they do not consider the loss of local model weights.

Table 3.4: Comparison on the critical communication cost of vehicles

Framework	PPDL [84]		SP-CIDS [90]	CRS	
	LWE-based encryption	Paillier encryption		*CH*	Member vehicles
Communication rounds	2	2	$2n_{\text{nb}}$ **	2 ††	2 ††
Propagation distance	Longer (To *CS*)	Longer (To *CS*)	Shorter (to neighbor vehicles)	Shorter (to the RSU)	Shorter (to *CH*)
Size ‡‡	29491 bytes	1024 bytes	2 bytes	1024 bytes	2 bytes
Required time §§	267.67ms	9.26ms	0.28ms	3.01ms	0.28ms

** n_{nb} *denotes the number of one-hop neighbor vehicles.*
†† *The communications in the cluster generation process are not counted.*
‡‡ *Size of the encrypted, perturbed or masked weight vectors.*
§§ *The end-to-end delay of each vehicle simulated with the corresponding weight vector size and the propagation distance in the scenario of countrywide data collection. Message signing and encryption are not considered here for comparison.*

calculated based on this assumption. We set the same V2V distance, the same V2I distance and the same RSU-to-server distance for all the frameworks. Vehicle mobility is not considered for simplicity and easy comparison. We omit the detailed analysis and calculation processes for brevity and refer readers to [84] and [90] for more details. Overall, in CRS, both the communication and computation costs of member vehicles, which take the largest proportion of the vehicles, are relatively low. Although the total computation cost is larger than that of SP-CIDS, and there are additional processes in CRS (i.e., cluster generation), CRS is more suitable for IoV considering the vehicle's overhead, the privacy preservation achievements and packet loss handling.

3.8 Discussion and Future Work

We further discuss the limitations, assumptions and future work:

Machine learning algorithms and optimization methods: the proposed CRS has good scalability in the aspects of machine learning algorithms and optimization methods. We adopt the SGD optimization method in this work to train

Table 3.5: Comparison on the critical computation cost of vehicles ($n = 20$)

Framework	PPDL [84]		SP-CIDS [90]	CRS	
	LWE-based encryption	Paillier encryption		*CH*	Member vehicles
Encryption	$1T_{\text{2l_mMUL}} + 2T_{\text{l_sMUL}} + 3T_{\text{l_ADD}} \approx 5338.91$ms	$1T_{\text{pn_EXP}} + 1T_{\text{pn2_EXP}} + 1T_{\text{pn2_MUL}} \approx 98.07$ms	NA	$1T_{\text{pn_EXP}} + 1T_{\text{pn2_EXP}} + 1T_{\text{pn2_MUL}} \approx 98.07$ms	NA
Decryption	$1T_{\text{2l_mMUL}} + 1T_{\text{l_ADD}} \approx 2668.64$ms	$T_{\text{pn2_EXP}} + 1T_{\text{pn_MUL}} \approx 73.17$ms	NA	NA	NA
Perturbation	NA	NA	$2T_{\text{Z_ADD}} \approx 0.01$ms	NA	NA
Masking	NA	NA	NA	$20T_{\text{Z\_ADD}} \approx 0.07$	$20T_{\text{Z_ADD}} \approx 0.07$ms
Loss handling	NA	NA	NA	$1T_{\text{GE}} \approx 0.0117$ms	$5T_{\text{Z_MUL}} + 3T_{\text{Z_ADD}} \approx 0.03$ms

machine learning models because it is popular and easy to understand. We also provide a detailed application instance with an HRNN model and SGD in Section 3.6.1. As a supplementary work, we validate the performance of the model with 2000 videos (where 80% are used for training and 20% for testing) from the Car Crash Dataset [8]. The global test accuracy is around 89.5%.

According to the concrete problems and the requirements, other latest or superior models and optimization methods (such as ADMM, the alternating direction method of multipliers) can be used in CRS easily. In the future, the convergence rate (or the number of iterations), the training efficiency and the training accuracy of different machine learning algorithms and optimization methods can be compared.

Message encoding and packet recovery: in this work, to recover the lost packets, we adopt the network coding technique and propose the message encoding algorithm. The method is a better choice in the application layer compared with a straightforward method, retransmission. The reasons are twofold, a) the introduced computation and communication costs are still acceptable, as analyzed in Section 3.7. b) The protocol is simpler because the data senders (i.e.,

the member vehicles) do not need to wait for a reply, i.e., an acknowledgment or a retransmission indicator, from the head. In other words, there is no need for feedback. This property has a significant advantage considering the changing clusters.

To further handle the packet loss, we can introduce beacons to indicate the loss and ask for retransmission when some packets cannot be recovered with our protocol. This is similar to the idea of hybrid automatic repeat request (HARQ). Another possible solution is to improve the masking algorithm. Thus, even with packet loss, the data can be gathered correctly. Bonawitz *et al.* [15] propose a double-masking algorithm to handle user dropping, which can be considered as local data loss. However, it is more complex and costly. Overall, it is a trade-off between the recovery rate and the overhead, which is a topic worth further studying.

Changing clusters: in this work, we consider the clusters to be stable. One thing that needs to mention is that the clusters only need to be stable during the process of secure data gathering in one updating round. To be specific, after sending the masked w_i to CH, the member vehicles do not need to wait for any replies or participate in the following processes in the current round. Thus, the member departure, the member join, and the CH change after the local gathering do not affect the machine learning process. Similarly, after handling the received w_i and sending w_{local} to the RSU, CH do not need to participate any following processes.

Based on the analysis and simulations in Section 3.7, the stable time required in CRS is around 1s for the cluster. As introduced in Section 3.2.2, the stable clustering techniques have been studied for more than ten years. The existing research results can well support the requirement of CRS. As a future work, we can further study how to handle the changes in clusters during the local data gathering process.

Message size and communication efficiency: compared with the end-to-end distributed machine learning framework, the "vehicle clusters-RSUs-server" architecture reduces the communication cost of vehicles. To further improve the communication efficiency, the following directions can be studied: a) the pruning technique can be adopted to reduce the size of the trained model or the

weight vector. For example, Imteaj *et al.* [49] adopt a sample-based pruning mechanism. The server collects a small portion of data (or requests it from workers, i.e., vehicles in IoV scenarios) and carries out an initial pruning while initializing a global model. The pruned model is shared with workers for weight updating. The authors observe that the pruning can reduce around 90% of the model size and the loss of accuracy is negligible. Jiang *et al.* [51] initially prune the model at a selected worker who has more computation resources (such as the vehicle belonging to the company that publishes the training task). After that, both the server and other workers are involved in the pruning. Overall, these works support that pruning the weight vector is feasible while maintaining the accuracy of the model. b) Another method is to reduce the number of ciphertexts. In this work, each entry of the weight vector should be encrypted separately by the threshold Paillier cryptosystem, so that the server can average each entry. In the future, we can adopt the batch mode of homomorphic encryption [16, 24, 124, 69]. It allows a batch of entries to be encrypted in one ciphertext without destroying the homomorphic property of each entry.

Real-world experiments: in this work, we have conducted simulations with both emulated and realistic scenarios. A potential future work is to realize the proposed framework, especially the reliable and secure data gathering protocol, on lightweight devices and real vehicles. The real-world experiments can study the concrete resource requirement and provide supports and guidance to the system implementation in industry.

3.9 Conclusion

In this chapter, we presented a two-layered distributed machine learning framework for IoV. It provides a new approach to the architecture of vehicular machine learning and the application of the related techniques. The framework has the following properties: a) the local inputs and local model weight vectors are protected against honest-but-curious entities; b) the privacy of vehicle identities and trajectories is preserved against both the RSUs and the central server; c) the packet loss is handled in the application layer to provide more reliable communications; d) with the "vehicle clusters-RSUs-server" architecture, the framework can naturally reduce the burden on vehicles. We conducted simulations with both emulated and realistic scenarios.

A comprehensive evaluation of the packet recovery performance, the communication and computation costs, and the security and privacy preservation performance was given. We also compared CRS with other two typical distributed machine learning frameworks. Results showed that CRS achieves more security and privacy goals and is more suitable for IoV. In addition, we provided an application instance and the prospects of the framework in industry.

The research directions presented in Section 3.8 hold significance for future exploration, but we mainly focus on addressing the security and privacy concerns in IoV data aggregation in this dissertation. Thus, in the next work, we target providing advanced security protection over the proposed two-layered architecture. This follow-up work is introduced in Chapter 4, along with the detailed research motivation.

Chapter 4

Schnorr Approval-based Secure and Privacy-preserving IoV Data Aggregation

In Chapter 3, we proposed a two-layered data aggregation architecture for the Internet of Vehicles (IoV). Although it maintains certain natural advantages over traditional end-to-end architectures, it still has limitations in terms of security protection: a) the adversarial behaviors of vehicles are not considered; b) the message verification within a cluster is not specified. In addition, in both EAirQ (introduced in Chapter 2) and CRS (introduced in Chapter 3), the privacy of vehicle identities and trajectories is preserved with pseudo-IDs, which requires additional work in pseudonym management.

As a follow-up work, we focus on tackling the challenges of secure data aggregation in IoV and mitigating the limitations of not only the existing works but also our previous works. In this chapter, we introduce a novel Schnorr approval-based IoV data aggregation framework over the two-layered architecture. We propose a novel concept, data approval, based on the Schnorr signature scheme. With the approval, the fake data injection attack carried out by a cluster head can be defended against. Invalid sub-approvals can be identified efficiently and a new average can be easily calculated. Both the identities and trajectories of vehicles are protected and there is no need for pseudonyms. The separation of liability is achieved as well. Compared to our previous works, the framework not only provides advanced security and privacy protection for data aggregation, but also addresses the limitations.

The frequently used notations in this chapter are summarized in Table 4.1 for reference.

Table 4.1: Notations in Chapter 4

Notation	Description
(skr, pkr)	The private and public key pair of an RSU
$\{v_i \mid i \in N_{\mathrm{v}}\}$	A cluster composed with n_{v} vehicles. $N_{\mathrm{v}} = \{1, 2, \ldots, n_{\mathrm{v}}\}$
CH	A cluster head
(sk_i, pk_i)	The private and public key pair of a member vehicle
CS	The cloud server
(sks, pks)	The private and public key pair of CS
(ska, pka)	The private and public key pair of the trusted authority
UID	ID of a data aggregation event
$\alpha_{i,j}$	The mask agreed between v_i and v_j
$data_i$ and c_i	The individual sensory data and the corresponding masked data of v_i
$\overline{data}$	The aggregated value (i.e., the average of all $data_i$) of a cluster
$v_{i_{\mathrm{re}}}$	An adversarial member vehicle with an index i_{re}
β_i	The reconstruction parameter generated by v_i
$appr_i$	The sub-approval of v_i
$appr$	The cluster approval
$\widetilde{pk}$	The aggregated public key of a cluster
$cre_{i_{\mathrm{CH}}}$	The credential of CH with an index i_{CH}
key_1 and key_2	Two session keys between CH and the RSU
L_{rc}	A list of records from all member vehicles

4.1 Introduction

In the Internet of Vehicles (IoV), vehicles equipped with various sensors and computing resources can provide many valuable data. It is not a brand new topic but gains ongoing interest in both industry and academia [123]. The sustained attention is driven by ongoing technological advancements, which give rise to many potential scenarios and applications. For example, in smart city, traveling vehicles can monitor the air pollution, noise level and humidity level at a fine granularity [113, 64]. With

federated learning, vehicles can perform as local workers, contributing their local training results to build a globally shared learning model for parking space estimation [43]. The speed of vehicles, directions, and road conditions are also invaluable for traffic management and autonomous driving [89, 23].

Protecting the sensory data, identities and trajectories is important. Although many solutions have been proposed, researchers never stop their efforts to enhance the security, efficiency, and practicability. Kong *et al.* [54] present a multi-dimensional data collection scheme for IoV. The modified Paillier cryptosystem is adopted to protect the sensory data and location information maintained in the reports, whereas the real identity of vehicles is exposed to the server. An adversary can further infer the trajectories of vehicles by linking the communication activities at different road segments. Li *et al.* [57] propose a safe and eco-friendly speed advisory system. Each vehicle encrypts the sensory data with an ElGamal commitment. However, to get the aggregation of the data, a discrete logarithm problem should be solved, which is time-consuming. To preserve the privacy, pseudonyms are used. Additional work is required for pseudonym management.

In our previous work **CRS**, which is introduced in Chapter 3 and published in the IEEE Internet of Things Journal [65], we propose a two-layered architecture for privacy-preserving data aggregation in federated learning. In CRS, a cloud server would like to aggregate sensory data from the vehicles via road-side units (RSUs). Different from traditional end-to-end architecture, most of the vehicles do not need to directly upload data to the server (i.e., do not need to directly communicate with RSUs): clusters are formed with multiple vehicles, and a cluster head is elected for each cluster to locally aggregate the data and perform as a gateway between the cluster and RSUs. As a result, both the communication cost of vehicles and risks of exposing data, vehicle identities and vehicle trajectories to RSUs and the server are reduced. To further preserve the data privacy, the technique of data masking is adopted, with which the cluster head can locally calculate the average of all the sensory data without acquiring the raw data. However, a *bad* cluster head may upload a fake average value. A newly emerged challenge in the two-layered architecture is ensuring the authenticity of the average. In addition, in our previous work, both EAirQ (introduced in Chapter 2) and CRS, the pseudonym technique is used to preserve the privacy of vehicle identities and trajectories, which requires additional management work.

To tackle the above challenges while avoiding adopting the typical techniques

which may bring high computation and communication costs to vehicles, in this chapter, we propose a novel **S**chnorr **A**pproval-based IoV **D**ata **A**ggregation framework, **SADA**, over the two-layered architecture. In SADA, a cluster head securely aggregates the sensory data in a cluster with a regional data collection scheme, named **CluCol** (**Clu**ster **Col**lection). It relays the locally aggregated result to a cloud server via RSUs anonymously with a novel Schnorr signature-based approval. To be specific, the main work and contributions are as follows:

- Instead of adopting the original masking technique as CRS, we propose a recoverable masking technique to preserve the privacy of sensory data. A significant advantage is that the technique enables a cluster to recover from an input invalidation attack. Different from differential privacy, it does not reduce the data accuracy. It is more lightweight for vehicles than the encryption-based solutions.

- We present a new concept, *data approval*, based on multi-signature schemes. It addresses the major limitation of CRS: it can prevent a cluster head from uploading fake aggregated data on behalf of the whole cluster. Invalid approvals can be detected fast with a binary tree-based pre-checking algorithm. We formally prove the security of the approval scheme in the Random Oracle Model (ROM).

- The identity and trajectory privacy of the cluster head is preserved with the cluster approval. Individual vehicles are hidden behind the cluster head. Thus, there is no need for updating or management of pseudonyms, which saves the computation and communication resources of vehicles.

- Compared with CRS, SADA provides advanced security protection and circumvents the substantial high overhead of pseudonym techniques. It also achieves the separation of liability among vehicles, which is a valuable advantage in the two-layered architecture. A misbehaved cluster head can be identified with a cluster key audit strategy.

In the rest of the chapter, we introduce the related work and cryptography tools in Sections 4.2 and 4.3, respectively. A system model and threat model are provided in Section 4.4. In Section 4.5, we briefly introduce the framework design, followed by a detailed description in Section 4.6. In Section 4.7, we analyze and evaluate

the performance of the work. Simulations have been conducted in ns-3 (Network Simulator-3) [80]. In Section 4.8, we compare, analyze and prove the security of SADA. A conclusion is given in Section 4.9.

4.2 Related Work

To preserve the privacy of the sensory data, one typical technique adopted in IoT is differential privacy where noise is added to the raw data. Raja *et al.* [90] propose a secure and private-collaborative intrusion detection system, SP-CIDS, for VANETs. To train a detection model in a distributed and privacy-preserving manner, vehicles perturb the local training results with Laplace noise. However, the accuracy of the aggregated value is affected. Similarly, Tang *et al.* [105] propose a health data aggregation scheme where each health center adds Laplace noise to the data. To further protect the perturbed data, the Boneh–Goh–Nissim (BGN) cryptosystem is adopted, which is additively homomorphic. The server can obtain the sum of the data from all health centers without decrypting the individual ciphertexts. Kong *et al.* [54] adopt the modified Paillier cryptosystem for privacy-preserving sensory data collection in VANETs, which also has the homomorphic property. Li *et al.* [56] improve the labeled homomorphic encryption (LabHE) cryptosystem to tackle the privacy challenge among the data owners, untrustworthy servers, and the data users in Industrial Internet of Things (IIoT). Although these homomorphic encryption algorithms are widely adopted because of the usability and precision of the aggregation result, the computation and communication costs are relatively high.

Bonawitz *et al.* [15] propose a pairwise masking technique for federated learning. It allows a server to compute the sums of model parameter updates from individual clients in a secure and efficient way. To handle the dropped-out clients, a threshold secret sharing scheme is adopted for mask recovery. Compared with differential privacy, this technique can maintain the precision of the data. Compared with homomorphic encryption, it is efficient in computing. Thus, we adopt the masking technique in CRS and propose a secure and reliable data gathering protocol based on network coding. In the work of this chapter, SADA, we still adopt this technique for secure data aggregation but improve it for bad vehicle exclusion. The introduction of the technique, its limitations, and our corresponding improvements can be found in Sections 4.3 and 4.6.1.

To preserve the identity and trajectory, pseudonym is a widely-accepted technique.

Lu *et al.* [115] propose a privacy-preserving authentication scheme for emergency traffic message transmission in VANETs. Each vehicle generates a pseudo identity with an unhackable tamper-proof device. In Eco-CASA [57], vehicles are allowed to register multiple anonymous key pairs but the authors do not provide details. In CRS, we adopt the work of Moni *et al.* [77]. In this work [77], the authors propose a secure authentication and message verification scheme for vehicular ad-hoc networks (VANETs). Regional trusted authorities issue pseudo-IDs to vehicles, which are used in all communications to preserve the privacy.

In CRS, to protect the identities and trajectories of vehicles from RSUs and the server, we propose the two-layered architecture. Member vehicles are hidden behind cluster heads so that are protected. As for cluster heads, we adopt the pseudo-ID-based solution for anonymous communication. A limitation is that the pseudonym technique requires additional work in the management and updating of the pseudo-IDs and cryptographic keys. Thus, in this work, first, we adopt the architecture in CRS to protect the member vehicles in clusters. Then, we try to preserve the privacy of cluster heads in a different way: data approval.

4.3 Preliminaries

In this section, we introduce the basic cryptography tools and techniques used in the proposed framework. We also briefly discuss the necessity and rationale for making revisions on them and the corresponding differences.

Masking technique: data masking allows a server to aggregate data from client parties in a secure way. With a naive additive masking algorithm, each pair of clients (u_i, u_j) agrees on a mask $\alpha_{i,j}$ with the Diffie-Hellman protocol. Each client can generate a masked value for its secrets with all the masks it obtains. The server can calculate the sum of all the secrets only with the masked values. Details can be found in Chapter 2.

To handle dropped clients during the sum calculation, Bonawitz *et al.* [15] adopt a threshold secret sharing scheme. Each Diffie-Hellman secret used for generating masks is divided into multiple shares, which are then distributed to different clients. The masks can be recalculated with a threshold number of shares. As a result, if a client drops before providing its masked value, the corresponding masks can be removed, and the sum of all the secrets can still

be correctly calculated. Although we do not target handling packet loss or client drop as what we achieved in CRS, we adopt the masking technique with Shamir's secret sharing to preserve data privacy and enable recovery from an input invalidation attack. Different from the original design [15], there is no need to recalculate masks, which saves the computation cost. Details are further given in Alg. 4.1.

Schnorr signatures: the Schnorr signature scheme is a simple and efficient digital signature scheme. The key-prefixed variant works as follows: the signer and verifier agree on a cyclic group $\mathbb{G}$ of prime order q with generator g. The public key of the signer is $y = g^x$ where $x \in \mathbb{Z}_q^\times$ is the private key. Given a message m, the signature is $(s = k + xe, R = g^k)$, where $k \xleftarrow{\S} \mathbb{Z}_q^\times$ ($\xleftarrow{\S}$ denotes randomly sample a value from a group) and $e = \mathrm{H}(y\|R\|m)$. The verifier calculates $e' = \mathrm{H}(y\|R\|m)$ and checks $g^s \stackrel{?}{=} Ry^{e'}$.

An advantage of the scheme is that it supports batch verification, which is faster than individually verifying each one. Suppose there are n_m signers. Each signer with an index i generates a signature (s_i, R_i) with its public key y_i on its message m_i. The verifier can verify all the signatures together by checking $\prod_{i=1}^{n_\mathrm{m}} g^{s_i a_i^1} \stackrel{?}{=} \prod_{i=1}^{n_\mathrm{m}} \{R_i y_i^{e'_i}\}^{a_i^1}$, where a_i^1 are random numbers and $e'_i = \mathrm{H}(y_i\|R_i\|m_i)$. The equation can be calculated faster by some mathematical computation algorithms (e.g., the Bos-Coster's algorithm).

Multi-signature: a Schnorr multi-signature scheme allows a group of signers to produce a joint signature on a common message. The size of the joint signature is exactly the same as a regular Schnorr signature. With an aggregated public key, a verifier can verify the joint signature as normal. There is no need to expose the individual keys. The naive scheme is vulnerable to the rogue-key attack, where the aggregator can generate a joint signature on behalf of all the signers by itself. In our work, we adopt a new rogue-key attack-resistant variant, MuSig [70]. However, it cannot directly satisfy the specific security and efficiency requirements in the two-layered architecture. Thus, we propose a new concept, data approval, and use MuSig as the foundation of it. We also call it *Schnorr approval*. In Section 4.6.2, we introduce the rationale and the minimal but intelligent modifications we made on MuSig.

4.4 System Model and Threat Model

4.4.1 System model

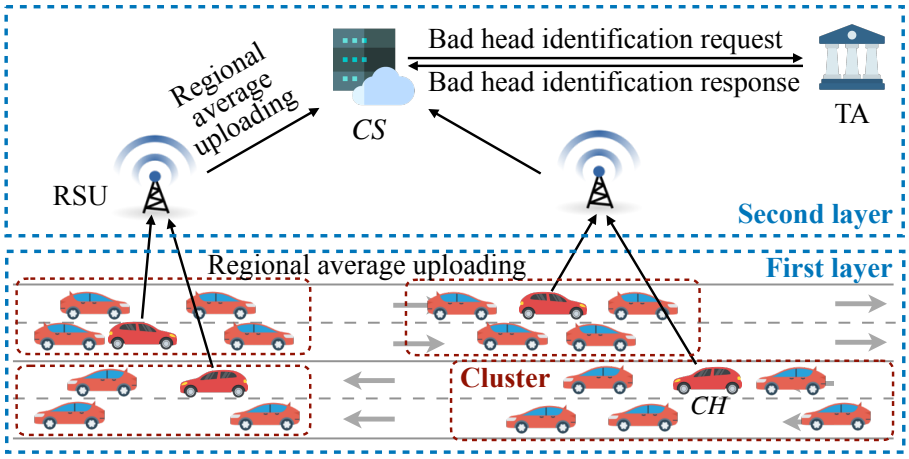


Figure 4.1: System model of SADA

As shown in Fig. 4.1, we formally define the entities:

- **Road-side units**: RSUs are base stations on the roadside. It relays the messages from vehicles to the server by wireless communication. The private and public key pair of an RSU is (skr, pkr).

- **Vehicle clusters**: a bunch of n_{v} vehicles traveling in the same direction, $\{v_i \mid i \in N_{\mathrm{v}}\}$, compose a *cluster*. $N_{\mathrm{v}} = \{1, 2, \ldots, n_{\mathrm{v}}\}$. Every vehicle keeps a sensory data, denoted by $data_i$. A **cluster head** CH performing as a gateway between the cluster and an RSU has the following tasks: a) to acquire data and verify the messages from member vehicles and calculate an average of the data; b) to pre-check the data approval; c) to identify bad member vehicles, and generate a new average and a new approval if needed; d) to send the average, approval and necessary parameters to RSUs. All the other vehicles are called **cluster members**. The main tasks of a member vehicle in each *sensing cycle* (i.e., a constant time slot) include generating a sub-approval, sending it and $data_i$ to CH, and helping CH in new average calculation when needed. The key pair of a member vehicle v_i is denoted by (sk_i, pk_i).

- **Cloud server**: the server CS (with a key pair (sks, pks)) runs in the cloud computing environment. The aggregated data are processed by CS in each

handling cycle. *CS* also works on identifying bad cluster heads with the help of a trusted authority.

- **Trusted authority (TA)**: the TA (with a key pair (ska, pka)) generates, distributes and manages the identities, credentials and cryptographic keys for all legal entities in the system. It has the responsibility to respond to bad head identification requests from *CS*.

4.4.2 Threat model

The entities who are not registered in the system are considered as external adversaries. They have the ability to capture messages transmitted between two parties. As for the registered entities, the assumptions are as follows:

- We assume the TA is fully trusted.
- RSUs and *CS* are honest-but-curious: they firmly follow the protocol but may be curious about the sensory data, vehicle identities and vehicle trajectories. Besides, to infer the vehicle trajectories, RSUs may collude with each other and link the communication activities of a vehicle of interest.
- Vehicles are considered as partial-honest-and-curious. All the vehicles are curious about the sensory data of others. In the partial-honest model, all vehicles are honest on their own individual sensory data, but may engage in specific adversarial behaviors: a) a *bad* member vehicle may provide an invalid sub-approval to *CH* which can lead to the invalidation of the aggregated data; b) a *bad CH* may upload a fake regional average, i.e., the cluster average of all the individual data. The assumptions are reasonable because the two roles (i.e., a member vehicle or *CH*) hold different rights and impacts: the average uploaded by *CH* representing all vehicles in the cluster has a higher impact than the individual data of one vehicle.

We summarize the major attacks we target within the threat model:

- **Data leakage attack**: the sensory data may be exposed to vehicles, RSUs, *CS* and external adversaries.
- **Identity leakage attack**: the real identities of vehicles may be exposed to RSUs, *CS* and external adversaries.

- **Trajectory tracking attack**: the trajectories of vehicles may be inferred by RSUs, *CS* and external adversaries.
- **Fake data injection attack**: a bad *CH* may try to upload a fake average value or fake approval to the RSU. In this work, even a trivial difference is considered as *fake*.
- **Input invalidation attack**: a *bad* member vehicle may provide an invalid sub-approval. It may also shift the accountability and potential punishment from the adversary to *CH*.
- **Eavesdropping attack**: the messages sent in the system may be captured by an adversary.
- **Message forgery and tempering attacks**: the messages may be forged or tempered.

In addition, we aim at tracing the bad vehicles and removing the negative impact of an attack as much as possible: a) *CS* should be able to not only detect the fake data injection attack, but also trace the bad *CH* who anonymously uploaded the fake average; b) when an input invalidation attack is detected, *CH* should be able to identify the bad member vehicle and exclude it from the protocol execution. A new regional average value without the sensory data of the bad vehicle should be calculated.

4.5 Framework Overview

The main protocol of the two-layered privacy-preserving data aggregation framework, **SADA**, is shown in Fig. 4.2. In this section, we briefly introduce the main idea. The notations, messages, algorithms and detailed protocols are introduced in Section 4.6.

The first layer of SADA is composed of vehicle clusters. All vehicles in the cluster can generate the same ID with a hash function $\mathrm{H}_{\mathrm{uid}}$, a pre-agreed timestamp tmp^1 and a uniquely encoded list of public keys of the vehicles L_{pk}:

$$UID = \mathrm{H}_{\mathrm{uid}}(L_{\mathrm{pk}} = \{pk_i \mid i \in N_{\mathrm{v}}\} \| tmp^1). \tag{4.1}$$

With binding tmp^1 and the information of all vehicles, we can use *UID* to uniquely

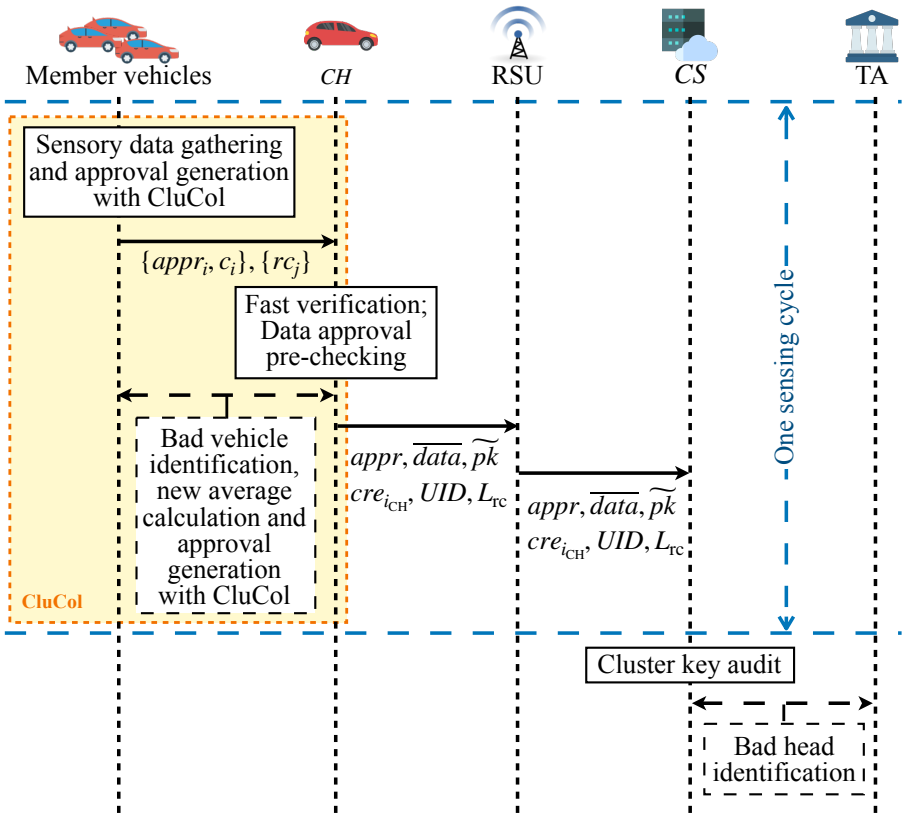


Figure 4.2: Protocol of SADA

identify a data aggregation event, i.e., aggregating data in a specific sensing cycle within a specific cluster.

We assume *CH* is first elected by all the vehicles in the cluster, which has been explored in our previous work CRS. In this process, a symmetric key $key_{v_{i,j}}$ and a mask $\alpha_{i,j}$ are exchanged and agreed upon by each vehicle pair (v_i, v_j) with the Diffie–Hellman key exchange protocol, as a preparation for regional data collection. The proposed lightweight and secure regional data collection scheme **CluCol** is specifically designed for vehicle clusters. It is the core component of SADA.

The design objective of CluCol is to allow *CH* to securely obtain $\overline{data}$, the average of all the data provided by every member vehicle in the cluster. To guarantee the confidentiality of $data_i$, v_i first masks it as c_i with the exchanged masks. An accurate sum can be calculated without acquiring any mask or $data_i$. This is the design in CRS. In SADA, the difference is that, instead of adopting the original masking technique, we propose a **recoverable masking protocol** integrating the masking technique with Shamir's secret sharing. When a bad member vehicle is identified, a new $\overline{data}$

can be easily calculated with the help of t_{sm} good member vehicles where t_{sm} is a threshold. There is no need to restart the process from the beginning. This design makes it possible for a cluster to recover from an input invalidation attack. We wish to emphasize that the goal and design of the integration of secret sharing is different from that in [15]. Details are given in Section 4.6.1.

In CRS, to aggregate the regional data, *CH* collects $\{c_i \mid i \in N_{\text{v}}\}$ and calculates the average $\overline{data}$ directly. Recall that we try to mitigate the limitation in CRS and assure the authenticity of $\overline{data}$. An intuitive design is to use the Schnorr multi-signature scheme, where each vehicle signs on $\overline{data}$ and *CH* calculates the joint signature. However, this design cannot meet all the requirements and defend against all the targeted attacks. *CH* may send a fake $\overline{data}$ to all member vehicles and obtain a valid joint signature. To defend against this fake data injection attack, we try to propose a new concept, approval, by adjusting a Schnorr multi-signature scheme, MuSig.

The approval involves the contribution of all v_i and confirms the authenticity of $\overline{data}$. In the proposed **data approval generation protocol**, different from the intuitive design, $\overline{data}$ is calculated by each v_i independently for authenticity. The commitments of $\{c_j \mid j \in N_{\text{v}} \wedge j \neq i\}$ are collected before sending c_i to other vehicles. Each vehicle generates a *sub-approval* $appr_i$ for $\overline{data}$. After collecting $appr_i$ from all member vehicles, *CH* generates the cluster approval *appr*. The details are given in Section 4.6.2.

We want to emphasize the necessity and rationale behind introducing the new concept. In a traditional multi-signature scheme, the joint signature is produced on a predetermined shared message, usually a transaction. Without all the required parties (i.e., signers), the transaction cannot be authorized and completed. Differently, with the proposed approval scheme, there is no transmission or a pre-agreed common message. All parties (i.e., vehicles) are responsible for calculating an average locally. The primary objective is to approve that the final average value is indeed calculated based on each party's individual data. It is specifically designed for the scenario of data aggregation within a vehicle cluster.

Another attack we target is the input invalidation attack. To defend against it, we present a **data approval pre-checking algorithm** in Section 4.6.3. With a thoughtful design, *CH* can check the validation of *appr* and efficiently trace the bad sub-approvals by maintaining intermediate results in a tree structure. The bad member vehicles can be excluded from the protocol execution with the proposed recoverable masking protocol.

To provide secure communication and fast verification in a cluster, the messages sent from member vehicles to *CH* are signed with the Schnorr signature scheme, which can be verified in batch as described in Section 4.3. If the verification fails, *CH* can identify the bad signatures with the binary search-based identification method. The main idea is to iteratively split the batch into two sub-batches and verify each of them separately. Besides, the sign-then-encrypt scheme is adopted to guarantee the confidentiality, authenticity and integrity of the messages.

As shown in Fig. 4.2, in the second layer, *CH* uploads both $\overline{data}$ and *appr* to an RSU, which are relayed to *CS* afterward. Other information sent from *CH* and the record $\{rc_j\}$ sent from cluster members are related to the **cluster key audit strategy**, which is designed to further detect the fake data injection attack. The details of the messages, the proposed strategy and the **bad head identification protocol** are provided in Sections 4.6.4 and 4.6.5.

In SADA, the **separation of liability** is achieved. To be specific, in the first layer, *CH* has the responsibility to aggregate an accountable regional average and identify the bad member vehicles (if any) with CluCol. Thus, in the second layer, both RSUs and *CS* assume the uploaded $\overline{data}$ and *appr* have been verified by *CH*. If *CS* further detects any invalid or fake *appr*, it is reasonable to believe that *CH* is an adversary. The separation of liability simplifies system management and facilitates the enforcement of potential punishment.

4.6 Framework Design

4.6.1 Recoverable masking protocol

Inspired by the work [15], we adopt a $(t_{\text{sm}}, n_{\text{v}} - 1)$ Shamir's secret sharing scheme in the recoverable masking protocol. Differently, the objective is to handle bad vehicles rather than dropped users. The revised design is given in Alg. 4.1: v_i first generates a construction parameter, β_i, and a hash of β_i, h_i. $n_{\text{v}} - 1$ shares are generated for β_i rather than the Diffie-Hellman secret (different from [15]). Both c_i and h_i are sent to an *executor*, who calculates the sum of all sensory data by summing all c_i. v_i distributes the shares $f_i(j)$ to the corresponding v_j. Note that, in CluCol, every vehicle performs as an executor. Parameter distribution and exchange are combined with the approval generation process (Section 4.6.2).

When a bad vehicle $v_{i_{\text{re}}}$ is identified, it is required to exclude the sensory data of

Algorithm 4.1. Recoverable masking algorithm

Input: sensory data $data_i$, p_{mk}, p_{sm}, N_{v}, masks $\{\alpha_{ij} \mid j \in N_{\text{v}} \wedge j \neq i\}$, threshold t_{sm}, and a hash function H_{mask}

1: v_i calculates the reconstruction parameter β_i as:

$$\beta_i = \sum_{j \in N_{\text{v}}: i<j} \alpha_{i,j} - \sum_{j \in N_{\text{v}}: i>j} \alpha_{j,i} \mod p_{\text{mk}}; \tag{4.2}$$

2: Generates the hash of β_i as $h_i = \text{H}_{\text{mask}}(\beta_i)$;

3: Picks $\{a_k^2 \xleftarrow{\S} \mathbb{GF}(p_{\text{sm}}) \mid k \in \{1, 2, \ldots, t_{\text{sm}} - 1\}\}$;

4: Generates $n_{\text{v}} - 1$ shares for $\forall j \in N_{\text{v}} \wedge j \neq i$ as

$$f_i(j) = \beta_i + \sum_{k=1}^{t_{\text{sm}}-1} a_k^2 j^k \mod p_{\text{sm}}; \tag{4.3}$$

5: Masks $data_i$ to c_i as

$$c_i = data_i + \beta_i \mod p_{\text{mk}}; \tag{4.4}$$

6: **return** c_i, h_i and $\{f_i(j) \mid j \in N_{\text{v}} \wedge j \neq i\}$

$v_{i_{\text{re}}}$ and get a new sum. To achieve this goal, *CH* is expected to reconstruct $\beta_{i_{\text{re}}}$ first as follows: *CH* picks t_{sm} vehicles except $v_{i_{\text{re}}}$. *CH* notifies every vehicle of the index i_{re} and requires the corresponding shares from the picked vehicles. With the t_{sm} shares, $\beta_{i_{\text{re}}}$ can be recovered by the Lagrange interpolating formula:

$$\beta'_{i_{\text{re}}} = \sum_{i \in L_{\text{sm}}} f_{i_{\text{re}}}(i) \prod_{j \in L_{\text{sm}} \wedge j \neq i} \frac{j}{j - i} \mod p_{\text{sm}}. \tag{4.5}$$

L_{sm} represents the index list of the picked t_{sm} vehicles. p_{sm} is a large prime. The reconstructed $\beta'_{i_{\text{re}}}$ is sent to each executor. If $\text{H}_{\text{mask}}(\beta'_{i_{\text{re}}}) = h_{i_{\text{re}}}$, suppose the masked value of $v_{i_{\text{re}}}$ is $c_{i_{\text{re}}}$, the executor can calculate a new sum sum_{new} from the old sum sum_{old} as follows:

$$sum_{\text{new}} = sum_{\text{old}} - c_{i_{\text{re}}} + \beta'_{i_{\text{re}}} \mod p_{\text{mk}} \tag{4.6}$$

where p_{mk} is an integer s.t. $data_i$ and $\sum_{i \in N_{\text{v}}} data_i \in \mathbb{Z}_{p_{\text{mk}}}$.

Different from the design in [15], there is no need to recalculate masks. $\beta'_{i_{\text{re}}}$ is only reconstructed once by *CH*. It obviously saves the computation cost of member vehicles.

4.6.2 Data approval generation

To resist against the fake data injection attack, we introduce a new idea, *data approval*. The principle is to intelligently make minimal modifications to MuSig and satisfy the requirements for the approval generation process.

As a preparation, each vehicle calculates $a_i^3 = \mathrm{H}_{\mathrm{agg}}(L_{\mathrm{pk}} \| pk_i)$ for every vehicle $\{v_i \mid i \in N_{\mathrm{v}}\}$ where $\mathrm{H}_{\mathrm{agg}}$ is a hash function. The aggregated public key is denoted as $\widetilde{pk} = \prod_{i=1}^{n_{\mathrm{v}}} pk_i^{a_i^3}$. Each v_i also selects $k_i \xleftarrow{\S} \mathbb{Z}_q^{\times}$ and calculates the nonce $R_i = g^{k_i}$. In MuSig, a signer (i.e., v_i) makes a commitment only on R_i. Differently, we require v_i to generate the commitment com_i as

$$com_i = \mathrm{H}_{\mathrm{com}}(R_i \| c_i \| h_i \| \{\mathrm{E}_{key_{\mathrm{v}_{i,j}}}(f_i(j)) \mid j \in N_{\mathrm{v}} \wedge j \neq i\}) \tag{4.7}$$

where c_i is the sensory data masked by Alg. 4.1. $\mathrm{H}_{\mathrm{com}}$ is a hash function. E denotes an encryption function. Each share $f_i(j)$ is encrypted with a symmetric key of (v_i, v_j). Instead of broadcasting com_i to all other vehicles in the cluster [70], v_i sends it to CH, who generates a list of the commitments as $L_{\mathrm{com}} = \{com_i \mid i \in N_{\mathrm{v}}\}$.

When and only when receiving L_{com} from CH, v_i sends a message m_i to the whole cluster as:

$$m_i := \{R_i \| c_i \| h_i \| \{\mathrm{E}_{key_{\mathrm{v}_{i,j}}}(f_i(j)) \mid j \in N_{\mathrm{v}} \wedge j \neq i\}\}. \tag{4.8}$$

After receiving $\{m_j \mid j \in N_{\mathrm{v}} \wedge j \neq i\}$ from all other vehicles, v_i first verifies $\mathrm{H}_{\mathrm{com}}(m_j) \stackrel{?}{=} com_j$ for all j. The verification checks the correctness of m_j and prevents an adversary from constructing a fake nonce [70]. v_i stores both $\mathrm{E}_{key_{\mathrm{v}_{j,i}}}(f_j(i))$ and h_j for potential usage in the recoverable masking protocol and discards other ciphertexts in every m_j: $\{\mathrm{E}_{key_{\mathrm{v}_{j,j'}}}(f_j(j')) \mid j' \in N_{\mathrm{v}} \wedge j' \neq j, i\}$. An aggregated nonce is generated as: $\widetilde{R} = \prod_{j=1}^{n_{\mathrm{v}}} R_j$.

Instead of directly providing each vehicle the average value that requires a Schnorr approval, we ask each v_i to calculate it by itself as:

$$\overline{data} = \frac{1}{n_{\mathrm{v}}} \sum_{j \in N_{\mathrm{v}}} c_j = \frac{1}{n_{\mathrm{v}}} \sum_{j \in N_{\mathrm{v}}} data_i. \tag{4.9}$$

It not only guarantees that no sensory data is exposed to other vehicles, but also confirms the authenticity of the masked data used. Each v_i now can generate a sub-approval on $\overline{data}$ as $appr_i = (s_i, \widetilde{R})$, where $s_i = k_i + a_i^3 sk_i e \mod q$ and $e = \mathrm{H}_{\mathrm{app}}(\widetilde{pk} \| \widetilde{R} \| \overline{data})$.

After collecting all the sub-approvals from member vehicles, *CH* generates the Schnorr approval for the cluster as $appr = (\widetilde{s} = \sum_{i \in N_\mathrm{v}} s_i \mod q, \widetilde{R})$. *appr* is generated with the contribution of all vehicles so that we can use it as an evidence of the data authenticity.

4.6.3 Data approval pre-checking

To defend against the input validation attack, *CH* verifies *appr* by checking $g^{\widetilde{s}} \stackrel{?}{=} \widetilde{R}\widetilde{pk}^{e'}$ where $e' = \mathrm{H}_{\mathrm{app}}(\widetilde{pk} \| \widetilde{R} \| \overline{data})$. We name it as the *first stage*. If *appr* is valid, *CH* can send it to *CS* on behalf of the whole cluster along with $\overline{data}$. If *appr* is invalid, *CH* has the responsibility to identify the invalid sub-approvals in the *second stage*, which can be achieved by verifying all $appr_i$ one by one. A shortcoming of this method is the considerable identification cost. Another choice is the binary search-based method: *CH* splits the index list $\{i \mid i \in N_\mathrm{v}\}$ to two sublists. For each sublist, L_{sub}, *CH* calculates $\widetilde{pk}_{L_{\mathrm{sub}}} = \prod_{i \in L_{\mathrm{sub}}} pk_i^{a_i^3}$, $\widetilde{R}_{L_{\mathrm{sub}}} = \sum_{i \in L_{\mathrm{sub}}} R_i$ and $\widetilde{s}_{L_{\mathrm{sub}}} = \sum_{i \in L_{\mathrm{sub}}} s_i$, and checks $g^{\widetilde{s}_{L_{\mathrm{nd}}}} \stackrel{?}{=} \widetilde{R}_{L_{\mathrm{nd}}} \widetilde{pk}_{L_{\mathrm{nd}}}^{e'}$. *CH* repeats the procedure until all the invalid sub-approvals are found.

However, the computation cost is still relatively high. Considering that *CH* cannot bypass the calculation of $\widetilde{pk}$, $\widetilde{s}$ and $\widetilde{R}$ in the first stage, we can maintain the intermediate results at the same time so that no additional calculation is required in both stages. To achieve this goal, we propose the approval pre-checking algorithm with tree structures. *CH* is expected to construct three binary trees while calculating $\widetilde{pk}$, $\widetilde{s}$ and $\widetilde{R}$: the *public key tree*, the *signature tree* and the *nonce tree*.

Taking the public key tree as an example, the values of the root node, the i-th leaf, and a parent node are $\widetilde{pk}$, $pk_i^{a_i^3}$, and the product of its two children, respectively. Suppose L_{nd} is the index list of all the leaves in the subtree of an internal node, we can calculate the value of the node as:

$$\widetilde{pk}_{L_{\mathrm{nd}}} = \widetilde{pk}_{\mathrm{left}} \cdot \widetilde{pk}_{\mathrm{right}} = \prod_{i \in L_{\mathrm{nd}}} pk_i^{a_i^3} \tag{4.10}$$

where $\widetilde{pk}_{\mathrm{left}}$ and $\widetilde{pk}_{\mathrm{right}}$ are the values of the left and right children, respectively. With Alg. 4.2, *CH* can construct the public key tree and obtain $\widetilde{pk}$ at the same time. Similarly, we can calculate the value of an internal node in the signature tree and nonce tree as $\widetilde{s}_{L_{\mathrm{nd}}} = \sum_{i \in L_{\mathrm{nd}}} s_i$ and $\widetilde{R}_{L_{\mathrm{nd}}} = \prod_{i \in L_{\mathrm{nd}}} R_i$, respectively. An example where $n_v = 10$ is shown in Fig. 4.3.

Algorithm 4.2. Public key tree generation algorithm

Input: n_{v}, $\{pk_i \mid i \in N_{\mathrm{v}}\}$ and $\{a_i^3 \mid i \in N_{\mathrm{v}}\}$

1: **procedure** CHILDREN(a parent node $node = (n_{\mathrm{lf}}, 0)$)
2: Generates two children nodes: the left child $node_{\mathrm{l}} = (\lfloor n_{\mathrm{lf}}/2 \rfloor, 0)$ and the right child $node_{\mathrm{r}} = (\lceil n_{\mathrm{lf}}/2 \rceil, 0)$;
3: **if** $\lfloor n_{\mathrm{lf}}/2 \rfloor ! = 1$ **then** CHILDREN $(node_{\mathrm{l}})$; **end if**
4: **if** $\lceil n_{\mathrm{lf}}/2 \rceil ! = 1$ **then** CHILDREN $(node_{\mathrm{r}})$; **end if**
5: **end procedure**
6: Defines the root node *root* with a pair $(n_{\mathrm{v}}, 0)$ where the first entry is the number of leaves and the second is the value of the node;
7: Generates the tree structure by CHILDREN $(root)$;
8: For $\forall i \in N_{\mathrm{v}}$, updates the i-th leaf as $(1, pk_i^{a_i^3})$;
9: Updates the values of all other nodes by (4.10);
10: **return** The public key tree, and the value of *root*

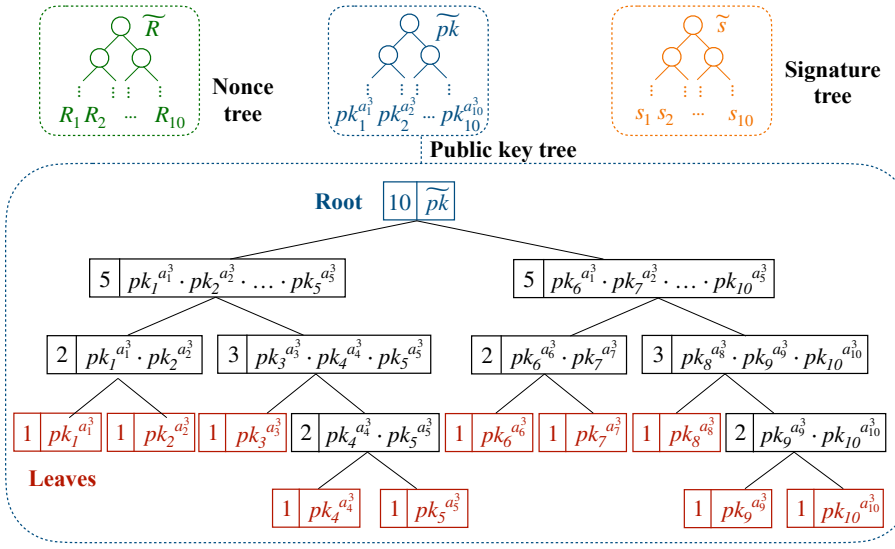


Figure 4.3: An example of the maintained trees with $n_{\mathrm{v}} = 10$

With the pre-constructed trees, we fully utilized the inevitable process in the first stage so that there is no need to calculate the intermediate aggregated values in the second stage, which saves the computation cost.

As described in Section 4.6.1, to exclude the bad vehicles who provided invalid sub-approvals, *CH* first notifies the cluster of L_{bad}, the index list of the bad vehicles, and L_{sm}, the index list of the randomly picked t_{sm} good vehicles:

$$m_{\mathrm{CH}}^1 := \{L_{\mathrm{bad}} \| L_{\mathrm{sm}}\}. \tag{4.11}$$

CH reconstructs $\{\beta'_{i_{\text{re}}} \mid i_{\text{re}} \in L_{\text{bad}}\}$ with the shares from the picked good vehicles and sends them back to all good vehicles. Then, every good vehicle can calculate the new regional average[1].

4.6.4 Regional data uploading

In SADA, each vehicle v_i maintains a credential issued by the TA as:

$$cre_i = \{\text{Sign}_{ska}(\text{H}_{\text{vid}}(\text{Commit}_{cka}(ID_i)), tmp^2) \| \text{Commit}_{cka}(ID_i) \| tmp^2\} \tag{4.12}$$

where $\text{Commit}_{cka}(ID_i)$ is a cryptography commitment on the identity of v_i with the commitment key of TA, *cka*. tmp^2 is the expiration time of the credential. H_{vid} is a hash function. The hash is signed with the private key of TA, *ska*. The credential is different from a traditional public key certificate. It does not reveal the identity or public key of v_i but can be used as a commitment to the identity.

A cluster head is expected to attach its credential $cre_{i_{\text{CH}}}$ when uploading reports. Beside $cre_{i_{\text{CH}}}$, as shown in Fig. 4.2, the approval *appr*, the average value $\overline{data}$, the aggregated key $\widetilde{pk}$, and the event ID *UID* form the report, which should be sent to *CS* through the RSU.

To upload the report securely, *CH* first sends two session keys, encrypted[2] with *pkr*, to the RSU: $\text{E}_{pkr}(key_1, key_2)$. Then, *CH* sends the following message to RSU:

$$m^2_{\text{CH}} := \text{E}_{key_1}(UID \| appr \| \text{E}_{pks}(\overline{data}) \| \widetilde{pk} \| cre_{i_{\text{CH}}} \| tmp^3) \tag{4.13}$$

where tmp^3 is a timestamp to defend against the message replay attack. $\overline{data}$ is encrypted with the public key of *CS* to provide further security. With *appr*, *CS* can verify the accountability of $\overline{data}$: it calculates $e' = \text{H}_{\text{app}}(\widetilde{pk} \| \widetilde{R} \| \overline{data})$ and checks $g^{\widetilde{s}} \stackrel{?}{=} \widetilde{R}\widetilde{pk}^{e'}$.

[1] A new approval should be generated for the new average. To provide further security, it is suggested to use different nonce R_i [70]. To save the communication cost, multiple R_i can be transmitted within m_i as a preparation in practice.

[2] For brevity, we do not distinguish the encryption algorithms represented by E. In practice, we can adopt the RSA algorithm to encrypt the session keys and $\overline{data}$ while the AES algorithm in m^2_{CH} and m^3_{CH} with key_1.

4.6.5 Cluster key audit and bad head identification

One possible attack from a bad *CH* is to generate a fake *appr* by itself without the participation of any member vehicle. The fake *appr* can be verified successfully by claiming $pk_{i_{\mathrm{CH}}}$ as $\widetilde{pk}$. This attack can be detected by *CS* with the proposed cluster key audit strategy and bad head identification protocol.

To achieve this goal, we require *CH* to collect a list of records from all member vehicles, L_{rc}, and upload it to the RSU. Suppose the total number of the records is n_{ps} and $N_{\mathrm{ps}} = \{1, 2, \ldots, n_{\mathrm{ps}}\}$. We formally define L_{rc} as follows:

$$L_{\mathrm{rc}} = \{rc_j = (\mathrm{H}_{\mathrm{vid}}(\widetilde{pk}_j), UID_j) \mid j \in N_{\mathrm{ps}}\}. \tag{4.14}$$

$\widetilde{pk}_j$ is the aggregated key generated and used in a past data aggregation event with an ID UID_j. $\mathrm{H}_{\mathrm{vid}}$ is a hash function. Each record rc_j in L_{rc} can be considered as an audit of cluster key $\widetilde{pk}_j$, corresponding to a past event UID_j and a previous cluster head CH_j. Note that during each sensing cycle, v_i only submits the records which have not been uploaded before (typically one record). Because L_{rc} is collected from all the vehicles in the cluster, it is impossible for RSUs or *CS* to infer the trajectory of *CH* by linking the events.

To upload L_{rc} securely, the message is designed as:

$$m^3_{\mathrm{CH}} := \mathrm{E}_{key_1}(\mathrm{HMAC}_{key_2}(L_{\mathrm{rc}}) \| L_{\mathrm{rc}}) \tag{4.15}$$

where HMAC is a hash-based message authentication code to provide authenticity and integrity.

In each handling cycle, *CS* compares the hash values in L_{rc}. If any record rc_j is invalid, according to the separation of liability principle, *CS* can believe that the corresponding CH_j is a bad vehicle. Thus, even if a bad head uploads a fake approval with the fake key in the current sensing cycle, it can be detected in the upcoming sensing cycles with the information provided by good vehicles. The real identity of a bad head can be revoked from $cre_{i_{\mathrm{CH}}}$ with the help of TA.

4.7 Performance Evaluation

In this section, we analyze and evaluate the computation and communication costs of SADA, and compare SADA with the latest related works [57, 54, 77, 15, 90],

including our previous work in Chapter 3, CRS [65]. Following CRS, we consider the scenario where vehicles travel in the same direction along a straight highway. As a representative, n_{v} is set to 20 for a cluster. t_{sm} is set to 10. All simulations are conducted 100 times to get an average result.

4.7.1 Computation cost

Table 4.2: Cryptographic operations and execution time [76] in Chapter 4

Operation	Abbr	Time (ms)
Scalar addition *	$T_{\mathrm{Z\_ADD}}$	0.0037
Scalar multiplication *	$T_{\mathrm{Z\_MUL}}$	0.0046
Scalar multiplication on $\mathbb{Z}_{p_{\mathrm{n}}}$ †	$T_{\mathrm{pn\_MUL}}$	1.5922
Scalar exponentiation on $\mathbb{Z}_{p_{\mathrm{n}}}$	$T_{\mathrm{pn\_EXP}}$	18.5283
Scalar exponentiation on $\mathbb{Z}_{p_{\mathrm{dh}}}$	$T_{\mathrm{pdh\_EXP}}$	26.9556
One-way hash function (SHA-256)	T_{SHA}	0.0088
Lagrange interpolation	T_{LI}	0.1404
Point addition on $\mathbb{G}$	$T_{\mathrm{G\_ADD}}$	0.0597
Point multiplication on $\mathbb{G}$	$T_{\mathrm{G\_MUL}}$	9.8134
Multiplication on $\mathbb{G}_{\mathrm{EG}}$ ‡	$T_{\mathrm{EG\_MUL}}$	0.1628
Exponentiation on $\mathbb{G}_{\mathrm{EG}}$	$T_{\mathrm{EG\_EXP}}$	20.2234
Logarithm on $\mathbb{G}_{\mathrm{EG}}$	$T_{\mathrm{EG\_LOG}}$	2813.8075
RSA-2048 encryption	$T_{\mathrm{RSA\_E}}$	2.8275
RSA-2048 decryption	$T_{\mathrm{RSA\_D}}$	568.1378
AES-256 decryption	$T_{\mathrm{AES\_DS}}$	0.0040
HMAC-SHA256	T_{HMAC}	0.0919
Tree construction	$T_{\mathrm{T\_C}}$	0.2917
Tree traversal	$T_{\mathrm{T\_T}}$	0.0015
Bos-Coster's algorithm	T_{BC}	170.4289

* *Related to* $\mathbb{Z}_{p_{mk}}$, $\mathbb{GF}(p_{sm})$, *and* $\mathbb{Z}_q$.
† $\mathbb{Z}_{p_n}$ *is the multiplicative group used in the Paillier cryptosystem.*
‡ $\mathbb{G}_{EG}$ *is a cyclic group with a 2048-bit prime order used for ElGamal commitment. Pollard's lambda method [76] is used to find discrete logarithms where the exponent is known to be small, e.g., 32 bits [57].*

The execution time of the cryptographic operations simulated with MIRACL [76] is listed in Table 4.2. The experimental platform is composed of an Intel Q9550

CPU with 2.83 GHz frequency, and 8GB RAM. We set $\mathbb{G}$, the group used in the Schnorr signature scheme, with the elliptic curve Secp256k1. A 256-bit order q and 2048-bit p_{dh} are chosen to provide the 128-bit security, where p_{dh} is the prime order of a Diffie-Hellman group $\mathbb{Z}_{p_{\text{dh}}}$. SHA-256 is adopted as the general one-way hash functions.

Table 4.3: Comparison on the computation cost of secure data aggregation

Scheme	Method	Data handling (ms)	Aggregation (ms) ¶	Recover (ms) ‖	Accuracy
SADA	Recoverable masking (in a cluster)	$(n_{\text{v}}-1+(n_{\text{v}}-1)t_{\text{sm}})T_{\text{Z\_ADD}}+1T_{\text{SHA}}+(n_{\text{v}}-1)(t_{\text{sm}}-1)T_{\text{Z\_MUL}} \approx 1.57$	$(n_{\text{v}}-1)T_{\text{Z\_ADD}} \approx 0.07$	*CH*: $1T_{\text{LI}}+2T_{\text{Z\_ADD}} \approx 0.15$; Member vehicle: $1T_{\text{AES\_DS}}+2T_{\text{Z\_ADD}} \approx 0.01$	✓
	RSA (RSUs-*CS*)	$1T_{\text{RSA\_E}} \approx 2.83$	$1T_{\text{RSA\_D}} \approx 568.14$	NA (Not applicable)	
Eco-CSAS [57]	ElGamal commitment	$3T_{\text{EG\_EXP}}+1T_{\text{EG\_MUL}} \approx 60.83$	$(n_{\text{v}}-1)T_{\text{EG\_MUL}}+1T_{\text{G\_LOG}} \approx 2816.90$	Not supported	✓
Kong's scheme [54]	Modified Paillier cryptosystem	$2T_{\text{pn\_MUL}}+2T_{\text{pn\_EXP}} \approx 40.24$	$2n_{\text{v}}T_{\text{pn\_MUL}}+1T_{\text{pn\_EXP}} \approx 82.22$	Not supported	✓
SP-CIDS [90]	Differential privacy	$2\ T_{\text{Z\_ADD}} \approx 0.01$	$(n_{\text{v}}-1)T_{\text{Z\_ADD}} \approx 0.07$	Not supported	✗
Original design [15]	Masking with threshold secret sharing	$(n_{\text{v}}-1+(n_{\text{v}}-1)t_{\text{sm}})T_{\text{Z\_ADD}}+(n_{\text{v}}-1)(t_{\text{sm}}-1)T_{\text{Z\_MUL}} \approx 1.56$	$(n_{\text{v}}-1)T_{\text{Z\_ADD}} \approx 0.07$	$1T_{\text{LI}}+(n_{\text{v}}-2)T_{\text{pdh\_EXP}}+n_{\text{v}}T_{\text{Z\_ADD}}+(n_{\text{v}}-1)T_{\text{SHA}} \approx 485.58$	✓

¶ *Operations or decryption required to aggregate the data of vehicles by CH, RSUs, or the server.*
‖ *Reconstruction process and new sum calculation, taking the scenario that one vehicle drops or should be removed as an example.*

To provide confidentiality on the sensory data, each vehicle masks the individual data in clusters. As shown in Table 4.3, all the data handling processes in Alg 4.1 take only 1.57ms. *CH* aggregates the shares within 0.07ms and further encrypts the average with the RSA algorithm in 2.83ms. Compared with the cryptography-based

methods [54, 57], SADA is more lightweight for vehicles. The heaviest task, i.e., the RSA decryption, is conducted by *CS* with cloud computing resources so that should not be a bottleneck. Although SP-CIDS [90] has a lower data handling cost than SADA, the accuracy of the aggregated data cannot be guaranteed. In SADA, *CH* takes 0.15ms to exclude the data of the bad vehicle and get a new sum with the proposed recoverable masking protocol. A member vehicle only needs 0.01ms to get a new sum. Compared with the works in [15], where each vehicle takes 485.58ms, our protocol is more lightweight.

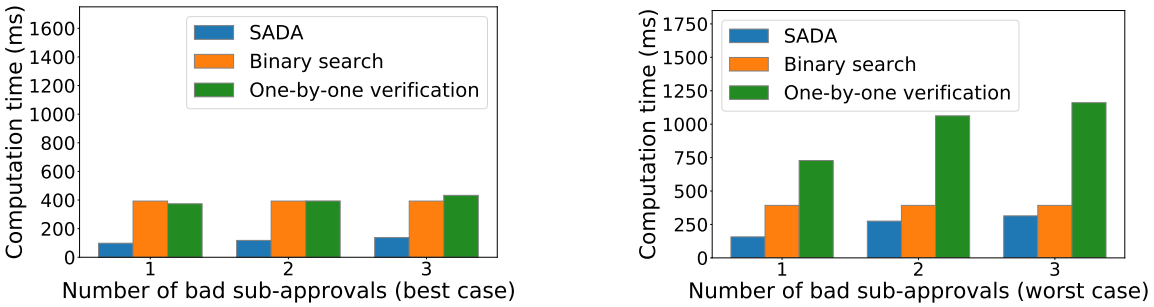


Figure 4.4: Comparison of the bad sub-approval identification

In data approval pre-checking, we propose an identification method with three binary trees. We compare it with the naive methods: a) the binary-search method without tree structures, and b) verifying the sub-approvals one by one. We set the number of bad vehicles $n_{\text{bad}} = 1, 2, 3$ considering that it should be much less than that of good ones in a cluster. As shown in Fig. 4.4, different from the first naive method, the tree structures save the $O(n_{\text{v}} \log n_{\text{v}})$ $T_{\text{Z\_ADD}}$ and $T_{\text{G\_ADD}}$ operations required for parameter aggregation. The second method always requires n_{v} verifications because *CH* does not have the knowledge of n_{bad}. Overall, the proposed method in SADA works better. We briefly introduce the space complexity of the design: there are $2n_{\text{v}} - 1$ nodes in each tree. The space requirement for the three tree structures and the node values are around 9.5KB in total, which is not a bottleneck for vehicles. It is worth noting that only *CH* is required to maintain the trees.

We adopt the Schnorr signatures with batch verification for efficient communication in clusters. As shown in Table 4.4, it requires low signing time and acceptable verification time. Although the message authentication code is more efficient, it is not suitable in our scenario because a) it cannot guarantee non-repudiation for messages, and b) additional work is required to distribute a symmetric key between each pair of vehicles. CRS does not consider message verification within a cluster but uses RSA

Table 4.4: Comparison on the cost of message verification in VANETs

Scheme	Method	Signing cost (for one message)	Verification cost (for $n_v - 1$ messages)	Signature size (bytes)	Security level (bits)
SADA	Schnorr-based batch verification	$1T_{G\_MUL} + 1T_{SHA} + 1T_{Z\_MUL} + 1T_{Z\_ADD} \approx 9.83$	$(n_v - 1)T_{SHA} + 1T_{BC} \approx 170.60$	48	128
Eco-CSAS [57]	Schnorr signature without batch verification	$1T_{G\_MUL} + 1T_{SHA} + 1T_{Z\_MUL} + 1T_{Z\_ADD} \approx 9.83$	$(n_v - 1)(2T_{G\_MUL} + 1T_{G\_ADD} + 1T_{SHA}) \approx 374.21$	48	128
Kong's scheme [54]	Message authentication code	$T_{HMAC} \approx 0.09$	$(n_v - 1)T_{HMAC} \approx 1.75$	32 **	256 **
Moni's scheme [77], and CRS [65]	RSA algorithm	$1T_{RSA\_D} \approx 568.14$	$(n_v - 1)T_{RSA\_E} \approx 53.72$	256	112

** *32 refers to the size of HMAC rather than a signature. HMAC-SHA256 provides 128-bit collision and 256-bit preimage resistance.*

Table 4.5: Comparison with CRS on message verification and identity privacy preservation

Scheme	Verification in cluster	Verification (*CH*-RSU communication)	Privacy of member vehicles	Privacy of *CH*
SADA	Schnorr-based batch verification	Achieved by Schnorr approvals	Protected by the two-layered architecture	Achieved by Schnorr approvals
CRS [65]	Not supported	RSA algorithm		Achieved by pseudo-IDs

signatures with pseudo-IDs for that between cluster heads and RSUs. Compared with SADA, CRS has fewer security guarantees and takes more time in pseudo-ID updating and RSA signing. We further compare these two frameworks in Tables 4.5 and 4.7.

To comprehensively evaluate the computation cost, in Table 4.6, we further summarize the computation time of all the algorithms and processes in SADA. Consid-

Table 4.6: Computation time of SADA

Algorithm or process	Time (ms)
Recoverable masking	1.57
Preparation for approval	207.38
Sub-approval generation	55.05
Approval generation and tree generation ††	0.97
Approval pre-checking ††	19.70
Invalid sub-approval identification ††	98.24
Reconstruction ††	0.14
New sum calculation	0.01
Regional data and records uploading ††	5.82

†† *Only performed by CH.*

ering that AES-256 is adopted to encrypt the messages transmitted in clusters, the total calculation time for a member vehicle and *CH* is both less than 700ms. If a bad member vehicle exists, to get a new sum and a new approval, around 115ms (a member vehicle) and 515ms (*CH*) are required, additionally and respectively. The results show that SADA is lightweight for vehicles.

4.7.2 Communication cost

To evaluate the communication cost, we conduct simulations in ns-3.34. The length of a cluster is set to 400m. The distance between *CH* and an RSU is in the range of 0m to 400m. We consider *CS* collects data from a city (e.g., Victoria) or a country (e.g., Canada). Thus the distance between an RSU to *CS* is 4.48km and 2757km, respectively. The standard of IEEE 802.11p is adopted for both V2V (vehicle-to-vehicle) and V2I (vehicle-to-infrastructure) communications. The main communication cost is analyzed and simulated in sequence as follows:

In a cluster, each v_i sends com_i to *CH*, which is 32 bytes with SHA-256. The returned list L_{com} is considered as $20 \times 32 = 640$ bytes. Considering that each share can be represented with 2 bytes and encrypted with AES-256, the length of m_i is around 370 bytes. Overall, there are $2(n_{\text{v}} - 1) + n_{\text{v}}(n_{\text{v}} - 1)$ messages sent in a cluster as a preparation for approval generation. Sending com_i, L_{com} and m_i takes 0.33ms, 1.93ms, and 0.79ms, respectively. Each sub-approval (48 bytes) is transmitted in 0.35ms.

When a bad sub-approval is detected, t_{sm} vehicles send the corresponding shares to *CH*. The reconstructed $\beta'_{i_{\mathrm{re}}}$ is 2 bytes. There is no need for vehicles to re-exchange masks where $\mathcal{O}(n_{\mathrm{v}})$ messages should be sent for each vehicle. With the proposed protocol, only $\mathcal{O}(1)$ messages are required between each vehicle and *CH*. Thus, the communication cost is low (around 0.31ms).

Recall that the communication in a cluster is protected with the Schnorr signature and AES-256. All the above results are simulated with considering the size of signatures and ciphertexts. We further compare the signature size in different VANETs-related works in Table 4.4. Overall, the V2V communication in a cluster is lightweight.

As for the V2I communication, considering *CH* traveling at 20m/s, it takes 0.56ms on average to transmit the session keys from *CH* to the RSU. Then, to upload the aggregated data and L_{rc}, two messages are sent from *CH*: m^2_{CH} (864 bytes) and m^3_{CH} (1024 bytes). They cost 2.14ms and 2.37ms, respectively. Suppose the same sign-then-encrypt scheme is adopted between RSUs and *CS*, the message sent takes 0.13ms and 56.33ms for citywide data aggregation and national data aggregation, respectively.

It is worth emphasizing that, different from the pseudonym-based schemes [77, 65], SADA does not require vehicles to acquire and update pseudonyms with TA, which further saves the communication cost for vehicles.

In this work, we do not consider cluster changes in a sensing cycle. We can further narrow the assumption down as: the cluster is stable in the process of regional data aggregation. The required time is around 1.5s (considering the presence of bad member vehicles) with the previous analysis and simulation. In this case, the assumption is reasonable in practice with the many existing techniques. We refer readers to Chapter 3 for more discussion about cluster stability.

4.8 Security Analysis

In Table 4.7, we further compare SADA with the related works focusing on secure data aggregation in VANETs. It shows that, considering both the security and privacy protection performance and the computation and communication costs, SADA is more suitable for IoV data aggregation. In the following subsections, we analyze the security protection and privacy preservation of SADA against the typical attacks and potential behaviors of the adversaries.

Table 4.7: Comparison on security and privacy protection

Property	SADA	CRS [65]	Kong's scheme [54]	SP-CIDS [90]	Eco-CSAS [57]
IoV data confidentiality	✓(Masking)	✓(Masking)	✓(Modified Paillier)	✓(Differential privacy)	✓(ElGamal commitment)
Identity privacy	✓(Cluster and approval)	✓(Cluster and pseudonym)	✗	✗	✓ (Pseudonym)
Trajectory privacy	✓(Cluster and approval)	✓(Cluster and pseudonym)	✓ ✗ ††	✗	✓ (Pseudonym)
IoV data authenticity	✓(Approval)	✗	✗	✗	✗
Approval validity	✓(CluCol)	NA	NA	NA	✓(Zero-knowledge proof) ‡‡
Malicious head or RSU detection §§	✓(CluCol)	✗	✗	✗	✓(Blockchain)
Message authentication and integrity	✓(Digital signature and HMAC)	✓✗(Digital signature only for *CH*-RSU communication)	✓(Message authentication code)	✗	✓(Digital signature)

†† *The location information maintained in the report is protected but the trajectory can be inferred by linking the communications.*
‡‡ *The scheme can prove that the ElGamal commitment is well-formed.*
§§ *In SADA we assume RSUs firmly follow the protocol but a malicious CH may perform a fake data injection attack. It can be prevented and detected by CluCol. In Eco-CSAS, there is no cluster but the authors consider that a malicious RSU may delete or tamper some critical data it collects. It can be detected with a blockchain where all the data are stored.*

4.8.1 Data leakage attack

The individual sensory data are masked with uniformly random masks, i.e., every mask is computationally indistinguishable from a uniformly sampled element. This property can be achieved by using a secure pseudorandom number generator [15]. With (4.2) and (4.4), Lemma 4.1 shows that the masked values $\{c_i \mid i \in N_v\}$ look

uniformly random as well [15, 103].

Lemma 4.1. *Fix* N_{v}, p_{dh}, p_{mk} *where* $p_{\mathrm{dh}} > p_{\mathrm{mk}}$, *and* $\{data_i \mid i \in N_{\mathrm{v}}\}$ *where* $\forall i \in N_{\mathrm{v}}$, $data_i \in \mathbb{Z}_{p_{\mathrm{mk}}}$. *Then,*

$$\begin{aligned} &\{\{\alpha_{i,j} \xleftarrow{\S} \mathbb{Z}_{p_{\mathrm{dh}}} \mid i < j\}, \{\alpha_{j,i} \xleftarrow{\S} \mathbb{Z}_{p_{\mathrm{dh}}} \mid i > j\} : \\ &\qquad \{data_i + \beta_i \mod p_{\mathrm{mk}} \mid i \in N_{\mathrm{v}}\}\} \\ &\equiv \{\{c_i \xleftarrow{\S} \mathbb{Z}_{p_{\mathrm{mk}}} \mid i \in N_{\mathrm{v}}\} : \{c_i \mid i \in N_{\mathrm{v}}\}\} \end{aligned} \tag{4.16}$$

where $\equiv$ *denotes the identical distribution.*

With Lemma 4.1, the probability that any adversary, who obtains any c_i of interest, infers the corresponding $data_i$ is equal to the probability imposed by its a-priori knowledge.

4.8.2 Identity leakage attack and trajectory tracking attack

In SADA, member vehicles are hidden behind *CH*. There is no direct communication between a member vehicle and the semi-honest *CS*. There is no need to share the identities or public keys of member vehicles to *CS*. In addition, for any adversary $\mathcal{A}$, inferring pk_i from $\widetilde{pk}$ is as hard as a random guess [70]. Thus, the identity and trajectory privacy of member vehicles are preserved.

The identity privacy of *CH* relies on the security of the commitment scheme. The Pedersen commitment scheme is perfect hiding in standard model with the discrete logarithm (DL) assumption: for $\forall i, i' \in \{1, 2, \ldots, n_{\mathrm{ID}}\}$ where n_{ID} is the number of registered IDs in the system, $P(ID_i = ID_{i_{\mathrm{CH}}} \mid \mathrm{Commit}_{cka}(ID_{i_{\mathrm{CH}}})) = P(ID'_i = ID_{i_{\mathrm{CH}}} \mid \mathrm{Commit}_{cka}(ID_{i_{\mathrm{CH}}}))$. Even for an all-powerful adversary, without the commitment key of TA, *cka*, the probability that it can infer i_{CH} from $\mathrm{Commit}_{cka}(ID_{i_{\mathrm{CH}}})$ is negligible larger than a blind guess. Besides, guessing $pk_{i_{\mathrm{CH}}}$ from the aggregated key is as hard as a random guess, which is meaningless. Thus, both RSUs and *CS* cannot infer the real identity or public key of *CH*.

The protection of the trajectory of *CH* is twofold: a) the roles of vehicles are changing between sensing cycles. The probability that *CH* is elected as the cluster head in next n_{c} continuous sensing cycles, is as low as $\frac{1}{{n_{\mathrm{v}}}^{n_{\mathrm{c}}}}$. It is hard for RSUs to trace the vehicle by linking the credential. b) In practice, *CH* updates $cre_{i_{\mathrm{CH}}}$ with TA after each use or a pre-determined time threshold, depending on the privacy requirement.

4.8.3 Fake data injection attack

To achieve the fake data injection attack, the intuitive way for the adversary, i.e., the bad *CH*, is to provide a fake $\overline{data}$ to the whole cluster and ask for a valid approval. This is defended against by the design: $\overline{data}$ is calculated by every vehicle rather than directly provided by *CH*.

Another potential method is *to forge a cluster approval on behalf of the whole cluster*. This attack is defended against because the approval scheme is provably secure under the DL assumption in the plain public-key model.

Theorem 4.1. *Assume there exists a CH which is a* $(T_{\text{fg}}, n_{\text{qs}}, n_{\text{qh}}, n_{\text{v}}, \epsilon)$ *forger against the approval generation algorithm with* p_{mk} *and group parameters* $(\mathbb{G}, q, g)$*, where* q *is* l_{q}*-bit long. Assume the hash functions* $\text{H}_{\text{com}}, \text{H}_{\text{agg}}, \text{H}_{\text{app}} : \{0,1\}^* \rightarrow \{0,1\}^{l_{\text{hash}}}$ *are modeled as random oracles. Then, there exists an algorithm* $\mathcal{C}$ *which* $(T_{\mathcal{C}}, acc(\mathcal{C}))$*-solves the DL problem for* $(\mathbb{G}, q, g)$*, with* $T_{\mathcal{C}} = 4T_{\text{fg}} + 4\mathcal{O}(n_{\text{v}}(n_{\text{qs}} + n_{\text{qh}})) + 4n_{\text{v}}T_{\text{G\_MUL}}$ *and* $acc(\mathcal{C}) \geq \frac{\epsilon^4}{(n_{\text{qs}}+n_{\text{qh}}+1)^2(n_{\text{qs}}+n_{\text{qh}})} - 8(\frac{p_{\text{mk}}+1}{p_{\text{mk}}})\frac{n_{\text{qs}}(n_{\text{v}}n_{\text{qs}}+n_{\text{qh}})}{2^{l_{\text{q}}}} - \frac{16(n_{\text{v}}n_{\text{qs}}+n_{\text{qh}})^2+3}{2^{l_{\text{hash}}}}$.

Theorem 4.1 indicates that if a bad *CH* in ROM runs in time at most T_{fg}, initiates at most n_{qs} signature protocols and at most n_{qh} hash queries to each of the random oracles, and forges a valid cluster approval on behalf of a cluster (n_{v} vehicles) with probability at least ϵ, then the corresponding DL problem can be solved in time $T_{\mathcal{C}}$ with probability at least $acc(\mathcal{C})$.

To prove Theorem 4.1, although we can follow the same strategy and use the double forking lemma as the original proof of MuSig [70], there are obvious changes in the detailed proof process. We provide the details in Appendix.

Now we discuss another method of the adversary to implement the attack: *CH* may *generate a fake but valid approval with its own public key* $pk_{i_{\text{CH}}}$ *and claim the key as the cluster key.* We define the security game as follows:

Security Game 4.1. *In a data aggregation event* UID_j*, the adversary generates an approval only with its own public key* $pk_{i_{\text{CH}}}$*. The adversary uploads the approval, following the protocol of SADA, but provides* $pk_{i_{\text{CH}}}$ *to CS instead of the real cluster key* $\widetilde{pk}_j$*. The adversary wins if CS cannot detect the fake approval with the cluster key audit strategy, i.e.,* $\text{Count}(EventA) < t_{\text{aud}}$*. The event EventA is true when CS receives a record corresponding to* UID_j *showing that* $\text{H}_{\text{vid}}(pk_{i_{CH}}) \neq \text{H}_{\text{vid}}(\widetilde{pk}_j)$*. The* Count *function counts the number of times that EventA = true.* t_{aud} *is a pre-defined system threshold which satisfies* $t_{\text{aud}} \geq 1$.

The adversary wins the game when and only when $pk_{i_{\mathrm{CH}}}$ occasionally leads to the same hash value of $\widetilde{pk}_j$. Thus, the security relies on the underlying hash function. We adopt SHA-256 which provides a 128-bit security level against collision and preimage attacks so that the probability that the adversary wins the game is negligible.

4.8.4 Input invalidation attack

The input validation attack can be detected with the proposed data approval prechecking algorithm. It is infeasible for an adversary to find an invalid sub-approval $(s_i', \widetilde{R})$ that satisfies $s_i' \neq s_i$ but $g^{(\sum_{j\in N_v \wedge j\neq i} s_j)+s_i'} = \widetilde{R}\widetilde{pk}^{e'}$. The proof is straightforward: $g^{(\sum_{j\in N_v \wedge j\neq i} s_j)+s_i'} = g^{(\sum_{j\in N_v \wedge j\neq i} s_j)+s_i}$ indicates that $s_i = s_i'$.

4.9 Conclusion

In this chapter, we proposed a Schnorr approval-based IoV data aggregation framework. A cluster head can aggregate the sensory data in a privacy-preserving way. Invalid sub-approvals can be identified efficiently and a new average can be easily calculated. With the approval, the authenticity of the aggregated data can be verified and there is no need for pseudonyms, which addresses the security and efficiency limitations of CRS [65]. Both the identities and trajectories of vehicles are protected. Compared with the related works, our work not only meets more security requirements but also is lightweight for vehicles.

Chapter 5

Flexible Non-interactive Short-term Implicit Certificate Generation for IoV

To preserve the privacy of vehicles in Internet of Vehicles (IoV), we adopted the pseudonym technique in Chapters 2 and 3. In Chapter 4, we tried to achieve it with a new approach and proposed a novel concept, data approval. While exploring new solutions holds considerable significance, it may require time for validation and deployment before being implemented in the real world. Thus, in addition to exploring the new directions, we also keep a close eye on the latest industry standards and try to make some improvements to the existing solutions.

A leading industry solution for secure and trusted communication in vehicular ad-hoc networks (VANETs) is the Security Credential Management System (SCMS). It uses anonymous certificates, functioning as pseudonyms, to preserve the privacy of vehicles. With the rapid development of advanced applications in IoV, such as crowdsensing and federated learning, vehicles need to communicate with each other or infrastructures more frequently (especially for the purpose of data aggregation), leading to a higher demand for pseudonyms. However, the current approach of certificate provisioning in SCMS is not able to fully support pseudonyms, due to storage limitation, cost of connectivity establishment and communication overhead of certificate downloading.

In this chapter, our objective is to tackle the challenge of certificate provisioning in SCMS, making the current industry solution better suit IoV. We propose a

non-interactive approach empowering vehicles to generate short-term key pairs and anonymous implicit certificates on their side. Our evaluation and comparison with previous work show that our solution not only effectively reduces the communication cost, but also grants vehicles greater flexibility in certificate generation and use. On the technical side, to the best of our knowledge, this is the first work which (1) applies *sanitizable signature* for non-interactive anonymous certificate generation, and (2) is specifically designed for SCMS, which opens up possibilities for extensions and applications in industry.

The frequently used notations in this chapter are summarized in Table 5.1 for reference.

Table 5.1: Notations in Chapter 5

Notation *	Description
(x, X)	The caterpillar key pair of a vehicle
$(\hat{x}, \hat{X})$	A cocoon key pair of a vehicle
(skv, pkv)	The signature key pair of a vehicle
(skv_j, pkv_j)	A short-term signature key pair of a vehicle
(skc, pkc)	The key pair of CA
$cert$	A CA-issued certificate associated with pkv
$cert_j$	A short-term certificate associated with pkv_j
(sks, pks)	The sanitization key pair issued by CA
(sks_j, pks_j)	A short-term sanitization key pair
rcv	The public key reconstruction value
h^1, h^2, h^2_j	Hash values
sig, sig^1, sig^2	Digital signatures
sig^2_j	The sanitized sig^2 with sks_j
$meta$	The metadata of a digital certificate
lv	The linkage value of a vehicle
slv_j	A short-term linkage value of a vehicle

* *In this chapter, all superscripts are notations (to distinguish the parameters of the same type) rather than exponents. All operations in the corresponding cryptography groups are expressed additionally. In all key pairs, the first entry is private key while the second one is public key.*

5.1 Introduction

Secure and trusted communication in vehicular ad-hoc networks (VANETs) is a topic of recent research interest [19, 9, 29, 106, 109]. One leading public key infrastructure (PKI)-based solution is the Security Credential Management System (SCMS) [19], which has been standardized by IEEE [46]. In SCMS, certificate authorities (CAs) generate and issue anonymous certificates to vehicles, which are used to verify the public keys of vehicles and provide message integrity and authenticity. The process is called *certificate provisioning*. Vehicles usually obtain a batch of certificates simultaneously and change pseudonyms (i.e., certificates and public keys) on demand.

As introduced in Section 1.1, the rapid development of the Internet of Things (IoV) techniques promotes advanced applications [34, 26, 63, 65, 116]. In crowdsensing, vehicles equipped with sensors can collect and upload real-time environmental data for specific tasks, such as traffic management [34], map updating [26] and air quality monitoring [63]. In federated learning, vehicles perform as workers to train models locally and exchange parameters with neighbor vehicles or road-side units (RSUs) [65]. Safety applications also have a promising future. For example, vehicles can collect real-time traffic information and broadcast emergency messages to avoid road accidents [116]. A noticeable situation is that, to facilitate these applications, the amount of vehicle-to-everything (V2X) communication is drastically increased. Considering that vehicles are recommended to change pseudonyms after a certain number of messages (e.g., every 100 messages [30]), these applications, in turn, urgently increase the demand for more certificates.

Two common solutions in SCMS to address this challenge are the following: (1) downloading more certificates from the CA (through RSUs or cellular networks) each time; (2) downloading certificates more frequently. However, these solutions may be problematic [19, 29]: a) storing a very large number of certificates may not be feasible due to the limited memory storage of vehicles; b) frequently establishing connectivity and downloading certificates are expensive considering the cost of RSU deployment and cellular network access; c) the high mobility of vehicles may lead to unreliable communication and considerable delays.

In addition, there is a lack of flexibility in the current design of SCMS. The system manager only defines the same certificate provisioning model (e.g., at least 20 certificates are valid simultaneously within one week [19]) for all the registered vehicles. Considering that a) vehicles may have different driving time and patterns

in real life; b) vehicles may have different privacy concerns; and c) communication requirements vary with different applications, an identical model is not sufficient and can lead to waste of certificates or lack of pseudonyms. Thus, a practical design should allow each vehicle to have its own personalized provisioning model.

To achieve the above desiderata, we propose a flexible **NO**n-**IN**teractive **S**hort-term certificate generation (NOINS) approach for SCMS. Following the recommendation in SCMS and a recent work, SIMPL [9], implicit certificates are adopted rather than the traditional explicit certificates, which is more communication-efficient. With this approach, vehicles can generate short-term implicit certificates from each CA-issued certificate, without interaction with the CA or RSUs. It not only reduces the communication rounds but also provides more flexibility in the generation and use of certificates. Our main contributions are as follows:

- We propose a new approach, NOINS, and non-trivially apply sanitizable signature, for generating flexible non-interactive short-term certificates. A vehicle can generate short-term pseudonyms by itself, which avoids frequently establishing connectivity or downloading certificates from CAs or RSUs. Compared with SCMS, SIMPL, and a recent work [2], we greatly reduce communication cost.

- Under our proposed approach, vehicles can personalize their certificate generation, e.g., change pseudonyms according to their driving habits and privacy requirements. This flexibility circumvents the issue of limited storage and prevents both the waste and the lack of certificates.

- The generated short-term public keys and certificates are unlinkable, providing vehicle privacy and message authentication. In addition, NOINS provides immutability, fraud-resistance and unforgeability.

- NOINS is designed on top of the well-accepted SCMS infrastructure, and can be directly integrated with it. There are many interesting problems that can be further explored in the new paradigm. We give an overview of these future directions in Section 5.7.

In the rest of the chapter, in Section 5.2, we introduce the related techniques and works. The system model and the threat model are given in Section 5.3. In Section 5.4, we present NOINS in details. In Section 5.5, we evaluate our work with theoretical

analysis and simulations. Our evaluation results show that our proposed approach achieves a clear reduction in communication cost for vehicles over previous work [19, 9, 2]. In Section 5.6, we analyze, prove and compare the security achievements. A discussion of interesting research problems in this new paradigm is given in Section 5.7, followed by a conclusion in Section 5.8.

5.2 Related Work

5.2.1 SCMS

SCMS, as a popular vehicular public key infrastructure (VPKI), received much attention these years. Typically, it adopts the elliptic curve Qu-Vanstone (ECQV) implicit certification model. This model involves scalar multiplication of EC points in public key verification but does not support pre-computing. Barreto *et al.* [9] propose an alternative for SCMS. The proposed Schnorr-based implicit certification (SIMPL) approach enables pre-computing and improves the efficiency, especially on the vehicles' side. The certificate provisioning process in SCMC with SIMPL is briefly introduced in Section 5.3.1 and given in Table 5.2.

In recent years, some researchers have worked on improving SCMS in different aspects. Sarker *et al.* [95] adopt the blockchain technique to advance the two functionalities, root certificate generation and the elector membership control, of the Elector-Based Root Management (EBRM) in SCMS. Noori *et al.* [79] suggest involving the spectrum misbehaviour reporting into SMCS to ensure the reliability of the wireless medium in VANETs. Gupta *et al.* [36] propose a V2V and V2I communication approach, where trusted cloudlets are introduced for trusted and reliable messages exchange, as a complement to SCMS. However, there is a lack of research on addressing the new challenges of certificate provisioning in SCMS to facilitate the advanced applications.

5.2.2 Sanitizable signature

One important underlying technique of NOINS is the sanitizable signature, which is first proposed by Ateniese *et al.* [6]. It allows an authorized party, called sanitizer, to modify the message signed by the signer but keeps the signature still valid. A typical way to achieve it is adopting chameleon hash functions [6]. However, due to the same

hash values, the sanitized messages of the same document can be linked, which is a crucial shortcoming of this method.

Brzuska *et al.* [20] first introduce the notion of unlinkability in sanitizable signatures, which is important to the design of NOINS. To achieve unlinkability, the authors adopt group signatures as the underlying signature scheme so that the sanitized messages have different signatures. Unfortunately, we cannot adopt it because a) having a large group of vehicles share the same group key leads to insecurity in certificate generation; and b) defining each vehicle and the CA as a group leads to the linkability of vehicles. Another line of works is based on re-randomized keys. Fleischhacker *et al.* [32] use a perfectly re-randomized secret key to sign the editable part of the message so that the sanitized messages, with different public keys, are unlinkable. However, the unchanged public key of the sanitizer should be revealed to prove the authenticity of the re-randomized public keys. Thus, it only guarantees that a sanitized message cannot be linked to an original message, but the messages belonging to the same sanitizer are still linkable.

In addition, sanitizable signature schemes are first proposed for database or document management, so that particular properties are usually taken into consideration [21, 17]. These properties, including invisibility, transparency and accountability, are not necessary for NOINS design but will increase the burden of the scheme. In summary, existing works on sanitizable signatures cannot be directly adopted for short-term certificate generation in VANETs.

5.2.3 Self-changing pseudonyms

There have been some efforts in self-changing the pseudonyms of vehicles. Qi *et al.* [87] propose a pseudonym-based certificateless authentication scheme. In this scheme, RSUs generate a list of partial keys for vehicles and periodically publish a related value, with which vehicles can generate key pairs on their side. Yang *et al.* [121] propose a self-agent pseudonym management scheme where vehicles generate short-term pseudonyms and send them to the server for activation. However, different from NOINS, these works are not truly non-interactive. Akil *et al.* [2] propose a truly non-interactive scheme based on non-interactive zero-knowledge proofs and Camenisch-Lysyanskaya (CL) signatures. It allows vehicles to broadcast messages with a self-updated pseudonym and a CA-issued attribute-based certificate. However, a) it does not support message encryption and signing; b) it has high commu-

nication and computation costs; and c) it cannot be integrated with SCMS directly. We further evaluate and compare our work with the most related works [19, 9, 2] in Section 5.6.

To the best of our knowledge, this is the first work applying (and further revising) sanitizable signatures in the non-interactive anonymous certificate generation problem and specifically designed for SCMS.

5.3 System Model and Threat Model

5.3.1 System model

As shown in Fig. 5.1, we formally define three entities:

- **Vehicles**: vehicles participating in concrete tasks (such as data collection in crowdsensing and model training in federated learning) have more V2X communications and high demand for anonymous certificates. They request implicit certificates *cert* from a certificate authority and generate short-term ones on demand. The initial key pair generated by a vehicle itself is (x, X), called caterpillar keys. The associated signature key pair of *cert*, used for V2X communication, is denoted as (skv, pkv).

- **CA**: the certificate authority is responsible for issuing implicit certificates *cert* to vehicles. The key pair of CA is denoted as (skc, pkc). Note that, there are different certificate authorities in SCMS. These authorities work together to ensure that no individual component knows the entire set of data that can be used to track a vehicle. For simplicity, in this chapter, we only use one CA as a logical component to represent the authorities in SCMS.

- **RSUs**: RSUs are wireless communication devices or base stations on the roadside. All the messages transmitted between vehicles and the CA are relayed by RSUs.

SCMS employs the unified butterfly key expansion process for certificate provisioning. To be specific, a vehicle first generates a caterpillar key pair $(x \xleftarrow{\$} \mathbb{Z}_q, X \leftarrow x \cdot g)$ where $\xleftarrow{\$}$ indicates randomly sample a value from a group. g is the generator of an elliptic curve group $\mathbb{G}$ of prime order q. Upon receiving certification requests from vehicles, a registration authority (RA) first expands every X to multiple $\hat{X}_i$

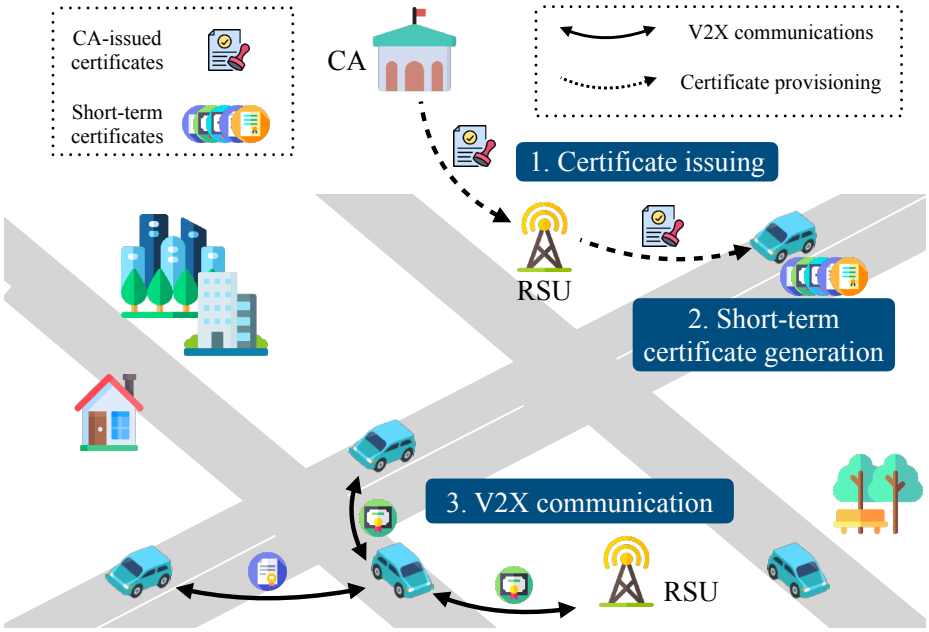


Figure 5.1: System model of NOINS

(called cocoon keys) with a function f and then shuffles all $\hat{X}_i$ from all vehicles. CA is responsible for issuing a certificate, with which the associated public key can be reconstructed, for each $\hat{X}_i$. This process allows each vehicle to request a batch of certificates (and key pairs) while avoiding CA from linking the vehicle with the issued certificate batch. The proposed solution, NOINS, works with the unified butterfly key expansion process but focuses on the processes afterwards, so that we omit the description of the related contents in the following sections. We refer readers to [19, 9] for more details. For simplicity, we leave out the subscript i and use $\hat{X}$ to denote a cocoon key of X. Note that all operations in group $\mathbb{G}$ and group $\mathbb{Z}_q$ are made (mod q) in this chapter.

5.3.2 Threat model

We assume each party in the system is either honest or corrupt, where a party is honest if it both follows the protocol and can be trusted with secret information. A corrupt party might be interested in, e.g., inferring the private key of other parties, pretending to be some legitimate vehicle, forging a certificate, deanonymizing other vehicles, or inferring their trajectories. We assume the CA is honest. Internal adversaries include RSUs and registered vehicles who are corrupt. There may exist external adversaries, who have the ability to capture and inspect the messages transmitted in the system

and certificates of other legitimate vehicles. The major security properties we consider are described as follows:

- **Immutability**: when generating short-term certificates from the CA-issued certificate, registered vehicles cannot modify metadata such as certificate expiration dates.
- **Anonymity**: certificates cannot be used to reveal vehicles' sensitive information or real identities.
- **Unlinkability**: short-term certificates and associated public keys should be unlinkable, i.e., adversaries cannot link two certificates (or two public keys) to the same vehicle and thus infer its trajectory.
- **Fraud-resistance**: an adversary can not pose as another vehicle, even with access to this legitimate vehicle's (legitimate) certificates.
- **Unforgeability**: external adversaries cannot forge a new valid certificate, even with access to valid certificates from legitimate vehicles.

Considering the applications in VANETs mentioned in Sections 5.1 and 5.3.1, we assume the vehicles who participate in these tasks would like to expose their real identities to CA to obtain task rewards. We further discuss this assumption in Section 5.7. Physical attacks, such as tracing a vehicle by its color or license plate, are out of scope.

5.4 NOINS Design

The flexible non-interactive short-term implicit certificate generation approach is shown in Table 5.2. For each cocoon public key $\hat{X}$, CA generates a reconstruction value *rcv* and an implicit certificate *cert*. With *rcv*, a signature public key, *pkv* can be reconstructed for the vehicle. To distribute the CA-issued certificate, a message m_{I2V} is sent to the vehicle. The message is encrypted with the vehicle's cocoon public key $\hat{X}$ so that only the vehicle can decrypt it. After obtaining and verifying the CA-issued certificate, the vehicle generates short-term certificates $cert_j$ with the sanitization technique. A new short-term key pair (skv_j, pkv_j) can be generated with each $cert_j$. For the sake of security and privacy, a short-term sanitization key pair (sks_j, pks_j) is used in this process. In a V2X communication, the receiver can verify

the authenticity of pks_j and then reconstruct pkv_j from $cert_j$. The verification of pkv_j will be implicitly conducted when it is used, i.e., when verifying a signature signed with the corresponding skv_j. Both the short-term sanitization public keys and the short-term signature public keys are unlinkable.

Our scheme allows for flexible, on-demand short-term certificate generation, which can e.g., be personalized to each vehicle based on their driving habits and privacy requirements. For example, a vehicle only driving on weekends typically need fewer certificates than a vehicle driving every day. A vehicle can pre-generate a batch of short-term ones and then generate new ones when suffering a shortage of pseudonyms. A vehicle uploading real-time sensory data needs more frequent pseudonym changing than a vehicle that uploads data less frequently.

The processes of certificate generation and provisioning in SCMS with traditional anonymous explicit certificates (explicit approach for short) and with implicit SIMPL are also given in Table 5.2 for easy comparison. Now we give a detailed description of the proposed NOINS approach.

5.4.1 CA-issued certificate generation

For each cocoon public key $\hat{X}$, CA is responsible for generating a certificate *cert*, which is valid in a certain period of time.

As shown in Table 5.2, CA first picks two random values r^1 and r^2 and calculates the reconstruction value *rcv* with $\hat{X}$. Besides *rcv*, *cert* contains some metadata *meta* (for example, the certificate format, the responsible authority, and the expiration time) and a linkage value *lv* as well. The system-defined metadata should not include any user-identifiable information. *lv* is a novel concept in SCMS and used for efficient revocation of certificate. It is unique in each CA-issued certificate. We refer readers to [19] for more details. Considering the applications of IoV, it can also be used to distribute task rewards to the corresponding vehicles.

Two hash values are generated with a hash function H. The private key of CA, *skc*, is used to generate sig^1 with the first hash h^1 so that no one can sanitize sig^1 without *skc*. It is considered as the immutable part.

The second hash h^2 is used in generating sig^2 with a sanitization private key *sks*. The sanitization key pair is generated by CA in advance as $(sks \in \mathbb{Z}_q, pks \leftarrow sks \cdot g)$. It is shared with the vehicle so that the vehicle can sanitize the editable part, i.e., sig^2, for generating new certificates. A key security problem is that, if each vehicle

Table 5.2: Certificate generation and provisioning process

	CA	$\rightarrow$ †	Vehicle	$\rightarrow$ ‡	Receiver
SCMS (explicit) [19, 9]	$r \xleftarrow{\$} \mathbb{Z}_q$; $pkv \leftarrow \hat{X} + r \cdot g$; $sig \leftarrow$ $\text{Sign}_{skc}(\{pkv, meta, lv\})$; $cert \leftarrow \{pkv, meta, sig\}$; $m_{\text{I2V}} \leftarrow$ $\text{Encry}_{\hat{X}}(\{cert, r\})$		$\hat{x} \leftarrow f(x)$; $\{cert, r\} \leftarrow$ $\text{Decry}_{\hat{x}}(m_{\text{I2V}})$; $\text{Veri}_{pkc}(cert)$; $skv \leftarrow \hat{x} + r$; $pkv \stackrel{?}{=} skv \cdot g$	$cert$; pkv	$\text{Veri}_{pkc}(cert)$
SCMS (SIMPL) [9]	$r \xleftarrow{\$} \mathbb{Z}_q$; $rcv \leftarrow \hat{X} + r \cdot g$; $cert \leftarrow \{rcv, meta, lv\}$; $h \leftarrow \text{H}(cert, pkc)$; $sig \leftarrow r + h \cdot skc$; $m_{\text{I2V}} \leftarrow$ $\text{Encry}_{\hat{X}}(\{cert, sig\})$		$\hat{x} \leftarrow f(x)$; $\{cert, sig\} \leftarrow$ $\text{Decry}_{\hat{x}}(m_{\text{I2V}})$; $h \leftarrow \text{H}(cert, pkc)$; $skv \leftarrow \hat{x} + sig$; $pkv \leftarrow$ $skv \cdot g \stackrel{?}{=} rcv + h \cdot pkc$	$cert$	$h \leftarrow \text{H}(cert, pkc)$; $pkv \leftarrow rcv + h \cdot pkc$;
NOINS	$r^1, r^2 \xleftarrow{\$} \mathbb{Z}_q$; $rcv \leftarrow \hat{X} + r^1 \cdot g + r^2 \cdot g$; $cert \leftarrow \{rcv, meta, lv\}$; $h^1 \leftarrow \text{H}(meta, pkc)$; $sig^1 \leftarrow r^1 + h^1 \cdot skc$; $h^2 \leftarrow \text{H}(rcv, lv, pks)$; $sig^2 \leftarrow r^2 + h^2 \cdot sks$; $m_{\text{I2V}} \leftarrow$ $\text{Encry}_{\hat{X}}(\{cert, sig^1, sig^2,$ $sks, r^2\})$	m_{I2V}	$\hat{x} \leftarrow f(x)$; $\{cert, sig^1, sig^2, sks, r^2\}$ $\leftarrow \text{Decry}_{\hat{x}}(m_{\text{I2V}})$; $(\hat{x} + sig^1 + sig^2) \cdot g \stackrel{?}{=} rcv +$ $h^1 \cdot pkc + h^2 \cdot pks$; To generate each $cert_j$: $slv_j \leftarrow \text{LinkGen}(lv)$; $r_j^3, r_j^4, \rho_j \xleftarrow{\$} \mathbb{Z}_q$; $rcv_j \leftarrow rcv + r^3 \cdot g$; $(sks_j, pks_j) \leftarrow$ $\text{RandKey}(sks, pks, \rho_j)$; $\{com_j, resp_j\} \leftarrow$ $\text{ProfGen}(r_j^4, \rho_j)$; $sig_j^2 \leftarrow$ $\text{Sant}(rcv_j, slv_j, sks_j,$ $pks_j, r^2)$; $cert_j \leftarrow$ $\{rcv_j, meta, slv_j\}$; $skv_j \leftarrow$ $\hat{x} + sig^1 + sig^2 + r_j^3$; $pkv_j \leftarrow skv_j \cdot g$	$cert_j$; pks_j; com_j; $resp_j$	$\text{ProfVeri}(com_j,$ $resp_j, pks_j, pks)$; $h^1 \leftarrow$ $\text{H}(meta, pkc)$; $h_j^2 \leftarrow$ $\text{H}(rcv_j, slv_j, pks_j)$; $pkv_j \leftarrow rcv_j +$ $h^1 \cdot pkc + h_j^2 \cdot pks_j$

† *The message sent from CA to a vehicle through an RSU.*
‡ *Values sent in V2X communications for authentication.*

has a unique sanitization key pair shared with CA, it should not be exposed to other vehicles, otherwise unlinkability is lost. In Section 5.4.2, we will show that this restriction cannot be achieved. To circumvent this, in NOINS, we consider the CA-

issued sanitization key pair to be identical to a large number of vehicles, for example, to all the vehicles registered in a city or all vehicles whose license plates share the same prefix. The principle is that revealing the sanitization public key would not reveal any sensitive information about a vehicle. An expiration date can be set for each sanitization key pair as well.

When issuing *cert* to the vehicle, the message $(\{cert, sig^1, sig^2, sks, r^2\})$ is encrypted with $\hat{X}$. It guarantees that even though the sanitization key is shared among vehicles, only the vehicle with the corresponding private key $\hat{x}$ can obtain r^2 and generate short-term certificates from *cert*. Encry is an asymmetric encryption function such as the elliptic curve integrated encryption scheme (ECIES) while Decry is the corresponding decryption function.

5.4.2 Short-term certificate generation

The vehicle first gets the cocoon private key $\hat{x}$ associated with $\hat{X}$, as mentioned in Section 5.3.1. After decrypting the message m_{I2V}, the vehicle can verify the CA-issued certificate *cert*. The correctness can be proved straightforwardly, as shown in (5.1).

$$\begin{aligned} &(\hat{x} + sig^1 + sig^2) \cdot g \\ &= \hat{X} + (r^1 + h^1 \cdot skc) \cdot g + (r^2 + h^2 \cdot sks) \cdot g \\ &= (\hat{X} + r^1 \cdot g + r^2 \cdot g) + h^1 \cdot skc \cdot g + h^2 \cdot sks \cdot g \\ &= rcv + h^1 \cdot pkc + h^2 \cdot pks. \end{aligned} \tag{5.1}$$

The vehicle can generate the short-term certificates from each *cert* on demand, which provides much flexibility.

Note that the linkage value *lv* is linkable if it is reused in different short-term $cert_j$ where $j \in \{1, 2, \cdots, n_{\text{cs}}\}$. n_{cs} is a system pre-defined upper limitation. Thus, the vehicle generates a short-term linkage value slv_j from *lv* for each $cert_j$. To achieve that, we adopt the same approach with SCMS (specifically, the last step of generating *lv* from pre-linkage values) [19] in Function 5.1. In short, it is a pseudorandom function (such as AES) in the Davies-Meyer mode. $\text{AES}_{lv}(a)$ denotes encrypting a with taking *lv* as the key. ID_{CA} is the identity string associated with CA. $[a]_{t_{\text{sv}}}$ denotes the t_{sv} significant bytes of bit-string a. For the sake of security, *lv*, as the secret key, should not be revealed to other entities.

The vehicle re-randomizes *rcv* with a random number r^3 to obtain a new recon-

$$slv_j \leftarrow [\mathrm{AES}_{lv}(ID_{\mathrm{CA}} \| j) \oplus (ID_{\mathrm{CA}} \| j)]_{t_{\mathrm{sv}}}$$
Output slv_j

Function 5.1: LinkGen(lv)

struction value rcv_j. We expect to use a sanitization key pair (sks, pks) to generate a new signature sig_j^2 for rcv_j and slv_j. The process is called *sanitizing*. Now we answer the previous question raised in Section 5.4.1, i.e., why we require CA to issue the same (sks, pks) to a great many vehicles. In a V2X communication, to reconstruct the short-term signature public key of a message sender, the message receiver must get pks. If pks is unique to the sender, the unlinkability is broken. Adversaries can link different $cert_j$ with the same pks.

There is another problem that must be considered: allowing different vehicles to use the same private sanitization key in certificate generation is not secure. To tackle the problem while upholding unlinkability, we do not directly use the CA-issued (sks, pks) in sanitizing. A unique sanitization key pair (sks_j, pks_j) will instead be generated from it and used in each $cert_j$ generation. With this design, each vehicle has its own short-term private sanitization keys and revoking pks_j does not break unlinkability. To this end, similar to the work in [32], the vehicle re-randomizes the shared sanitization key pair (sks, pks) by Function 5.2 [1].

$$sks_j \leftarrow sks + \rho_j$$
$$pks_j \leftarrow pks + \rho_j \cdot g$$
Output (sks_j, pks_j)

Function 5.2: RandKey(sks, pks, ρ_j)

Intuitively, the process of sharing (sks, pks) can be considered as CA distributes the power of short-term certificate generation to legitimate, registered vehicles. Then, the remaining task for the vehicle who uses (sks_j, pks_j) instead is to prove that they indeed get the power from CA. This can be achieved by a zero-knowledge proof of the relationship between (sks_j, pks_j) and (sks, pks). To be specific, the vehicle computes a commitment com_j and a response $resp_j$ by Function 5.3, which is a non-interactive version of the standard sigma (challenge-response) protocol [31]. Unlike [32], there is

[1] pks_j can also be calculated as $sks \cdot g$ but calculating $pks + \rho_j \cdot g$ is more efficient for the whole process. The result of $\rho_j \cdot g$ can be stored and used in the upcoming Function 5.3 so that one point multiplication operation on $\mathbb{G}$ is saved.

no need to expose the caterpillar public key X or cocoon public key $\hat{X}$ of the vehicle, which avoids the violation of unlinkability.

$$com_j \leftarrow r_j^4 \cdot g$$
$$cha_j \leftarrow \mathrm{H}(g, com_j, \rho_j \cdot g)$$
$$resp_j \leftarrow r_j^4 + cha_j \cdot \rho_j$$
Output com_j and $resp_j$

Function 5.3: ProfGen(r_j^4, ρ_j)

After setting up these keys, sig^2 is sanitized to sig_j^2 by Function 5.4. The main idea is that the vehicle can re-calculate a new sig^2 (i.e., sig_j^2) for a short-term $cert_j$ with different rcv, slv and pks (i.e., rcv_j, slv_j and pks_j). Thus, a short-term key pair (skv_j, pkv_j) can be generated and associated with $cert_j$. The security is guaranteed by the secrecy of r^2, r^3 and sks_j. The short-term certificate $cert_j$ contains the immutable part, $meta$, and the sanitized part, rcv_j and slv_j. It is still an implicit certificate so that pkv_j can be reconstructed from it and verified implicitly.

$$h_j^2 \leftarrow \mathrm{H}(rcv_j, slv_j, pks_j)$$
$$sig_j^2 \leftarrow r^2 + h_j^2 \cdot sks_j$$
Output sig_j^2

Function 5.4: Sant$(rcv_j, slv_j, sks_j, pks_j, r^2)$

5.4.3 Short-term certificate using in V2X communications

In a V2X communication, a vehicle, as a sender, signs its message m_{V2X} with its short-term private key sks_j. It needs to provide its certificate $cert_j$, pks_j, com_j and $resp_j$ to the message receiver for authentication.

The receiver first checks the expiration date of the certificate and the validity of pks. It then verifies the authenticity of pks_j by Function 5.5. If it outputs Succeed, the receiver generates h^1 and h_j^2 and reconstructs the short-term public key of the sender, i.e., pkv_j. The signed message m_{V2X} can be verified with pkv_j. Meanwhile, pkv_j is implicitly verified because a successful verification of m_{V2X} indicates that the sender indeed holds sks_j.

$cha_j \leftarrow \mathrm{H}(g, com_j, pks_j - pks)$
If $resp_j \cdot g = (pks_j - pks) \cdot cha_j + com_j$
Output Succeed
Otherwise, output Fail

Function 5.5: ProfVeri($com_j, resp_j, pks_j, pks$)

5.5 Performance Evaluation

In this section, following the evaluation paradigm of SIMPL [9], we compare the performance of NOINS, SIMPL and a recent non-interactive approach [2] (Akil's approach for short). SIMPL works better than the original implicit certificate-based approach of SCMS [9] so that we do not compare NOINS with the original implicit SCMS again. Considering that explicit certificates are still widely used in real life, we also involve the traditional explicit approach of SCMS for a clear comparison. We set $\mathbb{G}$ with the elliptic curve Secp256k1. A 256-bit prime order q is adopted to provide 128-bit security. We choose SHA-256 as the general one-way hash function. RSA-2048 with SHA-256, which is supported in the widely known X.509 certificates, is adopted as the digital signature scheme in the explicit approach. As recommended by SCMS, ECIES with a 256-bit curve is selected as the encryption function in m_{I2V}. The parameters used in Akil's approach are based on the recommendation of the idemix protocol [45].

We start by measuring the computation time. The experimental platform is composed of an Intel Q9550 CPU with 2.83GHz frequency, and 8GB RAM. Following the work in [9], multiplications by *pkc* are optimized by the Comb method (with a window width $w = 8$). Suppose that in both the explicit approach and SIMPL, n_{c} certificates are provided for a vehicle in total. Suppose there are n_{ci} CA-issued certificates in NOINS and Akil's approach. n_{cs} short-term certificates (or pseudonyms in Akil's approach) are generated from each of them and $n_{\mathrm{c}} = n_{\mathrm{ci}} \cdot n_{\mathrm{cs}}$. In other words, we consider the number of total certificates (or pseudonyms) a vehicle can use to be the same for comparison, no matter whether it is a CA-issued one or a self-generated short-term one.

The computation cost is given in Table 5.3 [2]. It can be observed that NOINS has

[2]This is the evaluation method we use: we comprehensively analyze all the operations on different cryptography groups in these approaches. We execute each operation 1000 times with the Multiprecision Integer and Rational Arithmetic Cryptographic Library (MIRACL) [76] for average results and then calculate the total computation time. For example, to generate one certificate, CA

Table 5.3: Computation cost for n_c certificates (in milliseconds)

	CA	Vehicle	Reciever
Explicit [19]	$307.76n_{\mathrm{c}}$	$11.84n_{\mathrm{c}}$	$1.98n_{\mathrm{c}}$
SIMPL [9]	$28.81n_{\mathrm{c}}$	$11.85n_{\mathrm{c}}$	$1.99n_{\mathrm{c}}$
Akil's approach [2] §	$3073.49n_{\mathrm{ci}}$	$3352.07n_{\mathrm{ci}}$ + $11553.26n_{\mathrm{c}}$	$1297.10n_{\mathrm{c}}$
NOINS	$37.37n_{\mathrm{ci}}$	$20.40n_{\mathrm{ci}}$+$34.10n_{\mathrm{c}}$	$27.61n_{\mathrm{c}}$

§ *We consider 10 attributes maintained in each certificate as an example. Because the approach is not comprehensively described in details by the authors, we only take the major processes into evaluation.*

the lowest computation burden on the side of CA (recall that $n_{\mathrm{ci}} < n_{\mathrm{c}}$). The explicit approach and SIMPL do not support self-generation of pseudonyms so that has lower computation burden on vehicles' side. NOINS and Akil's approach both hold this promising feature but NOINS is much more lightweight. Considering the increased computing capacity and the challenges of stable communication in VANETs, it is worth obtaining more improvements on communication burden with some sacrifice of computation efficiency. It echoes our design goal, i.e., reducing the communication cost and providing more flexibility for vehicles.

To measure the communication cost of vehicles, we conduct simulations with the Network Simulator (ns)-3.34 [80]. In all simulations, we model the CA-issued certificates as being sent to vehicles via RSUs. For completeness, the process of using certificates (i.e., sending a certificate and related parameters to a receiver in V2X communications) is also measured. We consider the scenarios of a) a small city (e.g., X, anonymized for double-blind review), and b) a large city (e.g., Y). It reflects different CA-to-RSU distances: around 5km and 60km, respectively. The propagation speed of the wired communication between CA and RSUs equals the speed of light, i.e., 299792.46km/s. The standard of IEEE 802.11p is adopted for V2X communication. The distance between a vehicle to the nearest RSU to be in the range of 0m to 300m. According to the two-second rule, we consider the distance between two vehicles is in the range of 10m to 100m in communication. The Transmission Control Protocol (TCP) is adopted for all communications.

We consider 16-byte metadata as an example. *slv* is set as 9 bytes. In the explicit

has to conduct one point addition (0.0573ms) and one point multiplication (8.4791ms) on $\mathbb{G}$, one RSA signing (278.9789ms), and one ECIES encryption (20.2469ms), which are 307.76ms in total. Please note that all computation time can be further reduced with advanced platforms or mature toolkits in practice.

Table 5.4: Communication time required for obtaining n_c certificates (in seconds)

	Small city			Large city		
n_c	500	1000	3000	500	1000	3000
Explicit [19]	0.49	0.98	2.95	0.65	1.31	3.93
SIMPL [9]	0.18	0.37	1.10	0.24	0.49	1.46
Akil's [2]	0.03	0.03	0.09	0.04	0.04	0.12
NOINS	0.01	0.01	0.04	0.02	0.02	0.05

Table 5.5: Total communication time for obtaining and using n_c certificates (in seconds)

	Small city			Large city		
n_c	500	1000	3000	500	1000	3000
Explicit [19]	0.83	1.65	4.96	0.99	1.98	5.94
SIMPL [9]	0.33	0.66	1.99	0.39	0.78	2.35
Akil's [2]	3.59	7.15	21.44	3.60	7.15	21.46
NOINS	0.23	0.44	1.31	0.23	0.44	1.33

approach and SIMPL, 9-byte *lv* is used as suggested by SCMS [19]. Differently, in NOINS, we set *lv* as 16 bytes to provide 128-bit security in AES. we consider $n_{cs} = 50$ and $n_c = \{500, 1000, 3000\}$ as representatives. We consider each time a batch of 20 CA-issued certificates is sent together to the vehicle in all simulations according to the CAR 2 CAR Communication Consortium (C2C-CC) model [13]. For simplicity, we omit the detailed description and directly give the results.

Table 5.4 shows the end-to-end delay required for obtaining n_c certificates (or pseudonyms in Akil's approach). Table 5.5 shows the total communication time for obtaining and using these certificates. It can be observed that our approach, NOINS, obviously saves the communication cost for vehicles. We further compare the security achievments in Section 5.6.

5.6 Security Analysis

In this section, we describe and analyze the security properties NOINS provides, based on the threat model defined in Section 5.3.2.

Immutability. We first define the property in NOINS as follows:

Definition 5.1. *The approach is immutable if there is no probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, i.e., a vehicle registered in the system and trying to generate short-term certificates, can win Security Game 5.1 with probability of success that is non-negligible in k where k is the security parameter.*

Security Game 5.1. *An adversary $\mathcal{A}$, with cacoon private key $\hat{x}$, obtains $\{cert, sig^1, sig^2, sks, r^2\}$ from CA where $sig^1 = r^1 + h^1 \cdot skc$. $\mathcal{A}$ wins if it can generate $sig^1_{\mathcal{A}}$ with respect to a bitstring of its choice $str_{\mathcal{A}}$ and satisfying $(\hat{x} + sig^1_{\mathcal{A}} + sig^2) \cdot g = rcv + h^1_{\mathcal{A}} \cdot pkc + h^2 \cdot pks$ where $h^1_{\mathcal{A}} = \mathrm{H}(str_{\mathcal{A}})$.*

Immutability is guaranteed with the security of sig^1 as stated in Theorem 5.1.

Theorem 5.1. *Immutability holds in NOINS if the generation of sig^1 is forgery-resistant.*

Proof. Suppose a PPT adversary $\mathcal{A}$ wins Security Game 5.1 with a non-negligible probability p. Then it holds that $(\hat{x} + sig^1_{\mathcal{A}} + sig^2) \cdot g = rcv + h^1_{\mathcal{A}} \cdot pkc + h^2 \cdot pks$, which leads to $sig^1_{\mathcal{A}} \cdot g = r^1 \cdot g + h^1_{\mathcal{A}} \cdot pkc$. It indicates that $\mathcal{A}$ is able to generate a forged signature for message $str_{\mathcal{A}}$ with the Schnorr signature scheme, with respect to public key pkc, with probability p. ■

With Lemma 5.1 [85] and Theorem 5.1, the immutability holds in NOINS.

Lemma 5.1. *The Schnorr signature is existentially unforgeable under chosen-message attacks (EU-CMA) in the Random Oracle Model (ROM) under the Discrete Logarithm (DL) assumption.*

Anonymity and unlinkability. Anonymity is guaranteed by adopting the same principle as SCMS, i.e., avoiding sensitive identity information in the metadata. We focus on discussing the unlinkability of the shared information in V2X communication: a) $cert_j = \{rcv_j, meta, slv_j\}$, b) pks_j, and c) com_j and $resp_j$.

Definition 5.2. *Two items of interest are unlikable if no PPT adversary $\mathcal{A}$ can win Security Game 5.2 with non-negligibly (in security parameter k) larger or smaller probability than the probability imposed by its a-priori knowledge* [3].

[3] A-priori knowledge [83] is the background knowledge adversaries have before running Security Game 5.2, such as the total number of registered vehicles.

Security Game 5.2. *An adversary $\mathcal{A}$ obtains two items a and b of interest. It applies a judge function $\{0,1\} \leftarrow \mathrm{Jug}(a,b)$ to determine if (i) a and b are generated from the same vehicle or with the same value, or (ii) a is generated from b. $\mathcal{A}$ wins if $\mathrm{Jug}(a,b)$ outputs the correct answer.*

Theorem 5.2. *Unlinkability holds in NOINS for any pair of distinct items from the set $\{cert_j, pks_j, com_j, resp_j\}$ and any pair of $\{cert_j, cert\}$ for $\forall j \in N_{\mathrm{cs}}$. $N_{\mathrm{cs}} = \{1, 2, \cdots, n_{\mathrm{cs}}\}$.*

Proof. Suppose the adversary $\mathcal{A}$ has a-priori knowledge K.

rcv_j is generated with a random variable r_j^3, which is picked uniformly from $\mathbb{Z}_q$. It implies that for any rcv, all rcv_j look uniformly random as well, as stated in Lemma 5.2.

Lemma 5.2. *Fix $\mathbb{Z}_q$, g and N_{cs}. For any rcv, we have*

$$\begin{aligned} &\{\{r_j^3 \xleftarrow{\$} \mathbb{Z}_q \mid j \in N_{\mathrm{cs}} : \{rcv + r_j^3 \cdot g \mid j \in N_{\mathrm{cs}}\} \\ &\quad \equiv \{\{rcv_j \xleftarrow{\$} \mathbb{Z}_q \mid j \in N_{\mathrm{cs}} : \{rcv_j \mid j \in N_{\mathrm{cs}}\} \end{aligned} \tag{5.2}$$

where $\equiv$ denotes the identical distribution.

Thus, except K, $\mathcal{A}$ cannot get any new knowledge helpful for $\mathrm{Jug}(rcv_j, rcv_{j'})$ or $\mathrm{Jug}(rcv_j, rcv)$ where $j, j' \in N_{\mathrm{cs}}$. Suppose K' is the a-posteriori knowledge of K, along with the two items of interest ($(rcv_j, rcv_{j'})$ or (rcv_j, rcv)), in Security Game 5.2. We have $|\Pr(\mathcal{A} \text{ wins} \mid K) - \Pr(\mathcal{A} \text{ wins} \mid K')| \leq \mathrm{negl}(k)$ where Pr is the probability of an event happens. negl is a negligible function.

A similar argument holds for slv_j, pks_j, com_j and $resp_j$: slv_j of the same vehicle is generated as a truncated value of AES cipher with the Davies-Meyer construction [19]. Similar with Lemma 5.2, every re-randomized key pair (sks_j, pks_j) with a uniformly chosen randomness has an identical distribution with (sks, pks) [32]. com_j and $resp_j$ are generated with uniformly and randomly picked r_j^4 and ρ_j. Thus, they all hold unlinkability.

■

Fraud-resistance. We first define fraud-resistance as follows:

Definition 5.3. *The proposed approach is fraud-resistant if for any PPT adversary $\mathcal{A}$, with key pair $(skv_{\mathcal{A}}, pkv_{\mathcal{A}})$, the success probability of winning Security Game 5.3 is negligible in k (security parameter).*

Security Game 5.3. *The adversary $\mathcal{A}$ has access to an oracle which outputs signatures $\theta = \mathrm{Sign}_{skv_j}(m_{\mathrm{V2X}})$ signed by a valid vehicle, with respect to the short-term public keys pkv_j and certificates $cert_j$ of that vehicle. $\mathcal{A}$ generates a signature $\theta_{\mathcal{A}}$ for a message str of its choice. Given a function* $\mathrm{Veri}_a(b, c)$ *which outputs 0 if the the signature b is not valid for the message c under the public key a and outputs 1 otherwise, $\mathcal{A}$ wins the game if* $\mathrm{Veri}_{pkv_j}(\theta_{\mathcal{A}}, str)$ *outputs 1.*

Based on Definition 5.3, the security of NOINS relies on the digital signature scheme adopted for communication in VANETs. We formally give the theorem and proof as follows.

Theorem 5.3. *The proposed approach is fraud-resistant if the digital signature scheme $\mathcal{DS}$, composed of a signing function* Sign *and a verification function* Veri*, used in V2X communication is forgery-resistant.*

Proof. The generated short-term certificate is used to reconstruct a public key. The public key is implicitly verified when the signature of the communicated message is verified. Thus, the certificate is only valid for the entity that has the associated private key.

If a PPT adversary $\mathcal{A}$, without the associated private key, can win Security Game 5.3 with a non-negligible probability of success p, it can forge signatures in $\mathcal{DS}$. ■

Unforgeability. We first define the property as follows.

Definition 5.4. *The short-term certificate satisfies unforgeability if any PPT adversary $\mathcal{A}$ (i.e., an unregistered vehicle) cannot win Security Game 5.4 with a non-negligible probability of success p in k (security parameter).*

Security Game 5.4. *An adversary $\mathcal{A}$ is not registered in the system but has access to an oracle $\mathcal{O}^1$ which outputs short-term certificates of legitimate vehicles, $cert_j = \{rcv_j, meta, slv_j\}$. A worse case is that $\mathcal{A}$ also has access to an oracle $\mathcal{O}^2$ which outputs valid sanitization key pairs $(sks_{\mathcal{A}}, pks_{\mathcal{A}})$ and the associated $com_{\mathcal{A}}$ and $resp_{\mathcal{A}}$.*

$\mathcal{A}$ wins if it can generate a certificate $cert_{\mathcal{A}}$ (with respect to some reconstruction value $rcv_{\mathcal{A}}$, some meta data $meta_{\mathcal{A}}$ and some sub-linkage value $slv_{\mathcal{A}}$ of its choice) and a key pair $(skv_{\mathcal{A}}, pkv_{\mathcal{A}})$ satisfying a) $pkv_{\mathcal{A}} = skv_{\mathcal{A}} \cdot g$, and b) $pkv_{\mathcal{A}} = rcv_{\mathcal{A}} + \mathrm{H}(meta, pkc) \cdot pkc + \mathrm{H}(rcv_{\mathcal{A}}, slv_{\mathcal{A}}, pks_{\mathcal{A}}) \cdot pks_{\mathcal{A}}$.

Theorem 5.4. *The proposed short-term certificate satisfies unforgeability with respect to Definition 5.4* [4].

Proof. An adversary $\mathcal{A}$ is able to generate $rcv_{\mathcal{A}}$ from existing rcv_j with any $r_{\mathcal{A}}^3$ it chooses. However, to get a short-term private key $skv_{\mathcal{A}}$ to win the game, both sig^1 and r^2 are required. If $\mathcal{A}$ wins the game with probability p, then it can make a forgery on the underlying scheme with probability p. With Lemma 5.1, the underlying scheme is provably secure, yielding a contradiction. ■

Comparison. NOINS is specifically designed for SCMS. It achieves the essential security and privacy requirements of SCMS and SIMPL. Differently, it can provide more flexibility and address the challenges of certificate provisioning.

Both NOINS and Akil's approach support non-interactive self-generated pseudonyms. They both provide anonymity, conditional anonymity/non-repudiation (i.e., CA can deanonymize vehicles), unlinkability, fraud-resistance, unforgeability, non-repudiation, and offline verification (i.e., a message can be verified without interacting with third parties) [2]. Differently, with Akil's approach, a vehicle can only have one pseudonym each time period, which can prevent Sybil attacks but has less flexibility. NOINS, following the recommendation of SCMS [19], allows a vehicle to hold multiple pseudonyms simultaneously based on its demands. In addition, Akil's approach only generates pseudonyms but does not change the key pair of a vehicle. Unlinkable message encryption and PKI-based message signing are not supported, which limits its application in VANETs. Our approach, NOINS, not only supports these important functions but is also much more lightweight than Akil's approach.

5.7 Discussion

As a new paradigm in SCMS, there are many interesting problems that can be further studied.

Limiting the power of vehicles: the research on the underlying technique, sanitizable signature, provides many possibilities for an enhanced design of NOINS.

[4]For unforgeability, we assume adversaries do not collude (i.e., share information with each other). Collusion trivially breaks unforgeability because an adversary may generate a new certificate with the secret information, such as sig^1 shared by another vehicle. As for the other security properties, collusion between two adversaries cannot bring any advantages for winning the corresponding game. See Section 5.7 for future directions on relaxing the non-collusion assumption.

One extension is to limit the power of sanitizers. For example, limiting the number of versions that they can generate on one document [22], i.e., allowing us to impose limitations on the number of pseudonyms a vehicle can generate with each CA-issued certificate. One possible method is to adopt the technique of cryptographic accumulator. A version number is attached to each short-term certificate. If any version number is used twice, a secret value (such as the private key) of the vehicle can be inferred. The remaining challenge is how to avoid the exposure of linkable information in this process, which may violate the property of unlinkability.

Malicious vehicles and collusion attacks: another valuable research direction is to consider fully malicious (i.e., arbitrarily deviates from the protocol) instead of semi-honest vehicles. For example, one assumption in this work is that vehicles would like to provide their real identities (i.e., the real slv_j which can be linked to identities according to the design in [19]) to CA for task rewards. Although it is reasonable for most real-world users, finding a solution to enforce it can provide more security, especially when we consider an adversary who does not care about rewards but only aims at taking some illegitimate actions and escaping from identity tracing.

Moreover, while we assume non-collusion for unforgeability, an interesting question is whether this property is possible with limited collusion. We know, for example, unforgeability holds if adversaries only share public information with each other.

Templates for personalized models: NOINS provides much flexibility for vehicles. Vehicles can work within various personalized models for certificate generation and use. Although there is actually no standard, a system manager can provide some templates for vehicles. This idea is widely used in industry; for example, computer security software companies (such as CORTEX) usually provide playbooks for users in security orchestration and automation. With these templates, we not only allow users to personalize their own models but also make the solution more user-friendly.

We give some template examples from different perspectives: a) a vehicle can generate enough short-term certificates with the home network before traveling outside. It can treat all the certificates as one-time-use pseudonyms and discard

them after use. This is suitable for a vehicle with good storage capability and strong concerns of network security and privacy. b) In a crowdsensing task of air quality monitoring [63], vehicles are asked to upload the observed air quality to the nearest RSU for the streets they passed by. Each uploading cycle is 15 minutes. Considering the risk of exposing their trajectories in this process, vehicles should change their pseudonyms every 15 minutes as well. c) According to the recommendation of the ETSI standard [30], vehicles can change pseudonyms every 5 minutes, 100 messages, or every 500m. It is more suitable when the communication is intensive and the privacy concern is not as strong. Defining different templates should take many concrete factors into consideration and has many possibilities.

Overall, applying the idea of sanitizable signatures in non-interactive short-term implicit certificate generation is a novel paradigm for SCMS, with obvious advantages in communication efficiency and flexibility. Interesting research problems can be further explored.

5.8 Conclusion

In this chapter, we proposed a flexible non-interactive short-term implicit certificate generation approach. After obtaining CA-issued certificates, a vehicle can generate short-term key pairs and certificates on demand. It is achieved by adopting and modifying the technique of sanitizable signature. Vehicles can determine the time of generation, the number of certificates, and the frequency of pseudonym changing. It not only avoids the waste or lack of pseudonyms but also significantly reduces the communication cost on vehicles' side. The proposed approach is designed on top of SCMS so that is naturally suitable for it. As a new paradigm, there are many extensions and applications to be explored.

Chapter 6

Conclusion

In this dissertation, we made efforts to address the security and privacy issues in Internet of Vehicles (IoV) data aggregation. We first focused on the application of air quality monitoring because air pollution has become a global concern and IoV makes it possible for a fine-grained updating of the air quality. The proposed framework, EAirQ, not only solves the sparsity problem in truth discovery but also preserves the privacy of the sensory data, vehicle identities and trajectories. This part of the work has been published in the *16th International Conference on Mobility, Sensing and Networking* (MSN 2020) [63] and *Digital Communications and Networks* (DCN 2023) [64].

Another promising scenario in IoV is distributed machine learning. Instead of adopting the traditional end-to-end data aggregation architecture, we specifically designed a two-layered architecture for IoV. The architecture uniquely involves vehicle clusters, road-side units, and a central server, which provides a basic guarantee of vehicle privacy while limiting the overhead. The proposed privacy-preserving distributed machine learning framework, CRS, can securely aggregate the local training results while preserving the privacy of vehicles. This part of the work has been published in the *IEEE Internet of Things Journal* (IoTJ 2024) [65].

To further enhance the security protection in the two-layered architecture, we introduced a novel Schnorr approval-based IoV data aggregation framework, SADA. In this framework, the fake data injection attack carried out by a cluster head can be defended against. The separation of liability is achieved as well. The proposed new concept, data approval, not only addresses the major limitations of CRS, but also achieves low-cost privacy preservation for vehicle identities in IoV data aggregation. It is a novel lightweight solution specifically designed for the two-layered IoV data

aggregation architecture. This part of the work has been submitted to the *IEEE Transactions on Vehicular Technology* [66]. In the future, more concrete attacks and security properties can be further explored in this new direction.

We not only aim at proposing new solutions, but also keep a close eye on the latest industry standards. This part of the work solves the remaining limitations of a leading industry solution, SCMS, and makes it more suitable for IoV data aggregation scenarios. We proposed a flexible non-interactive short-term implicit certificate generation approach, NOINS. It not only avoids the waste or lack of pseudonyms but also significantly reduces the communication cost on vehicles' side. This part of the work is to be submitted to the *ACM Transactions on Privacy and Security* [62]. Applying and further adjusting the idea of sanitizable signatures in non-interactive short-term implicit certificate generation is a novel paradigm for SCMS. Many extensions and applications can be explored, such as further limiting the power of vehicles in short-term certificate generation, and designing templates for personalized certificate provisioning models.

Another possible research direction is to consider the security of the authority. The authority is considered honest and trustworthy in our work. In practice, this can be achieved by dividing the role of a single authority into multiple ones. Even if one or some of them are corrupt, the secret values of users (i.e., vehicles) in the system will not be exposed. This is also the principle of SCMS. In addition, some techniques, such as blockchain, can be adopted to improve security or remove the requirement of a central authority.

Overall, in this dissertation, the major objective is to protect data security and preserve the privacy of vehicle identities and trajectories in data aggregation, while keeping both the communication and computation costs of vehicles low. We not only explored new technical solutions but also put efforts into improving the latest industry solutions. We proposed some new concepts and paradigms, which open up possibilities for research extensions and applications in industry.

Bibliography

[1] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella-Béguelin, and Paul Zimmermann. Imperfect forward secrecy: How Diffie-Hellman fails in practice. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 5–17, 2015.

[2] Mahdi Akil, Leonardo Martucci, and Jaap-Henk Hoepman. Non-interactive privacy-preserving sybil-free authentication scheme in VANETs. In *Network and DistributedSystem Security (NDSS) Symposium*, 2023.

[3] Ranwa Al Mallah, Alejandro Quintero, and Bilal Farooq. Distributed classification of urban congestion using VANET. *IEEE Transactions on Intelligent Transportation Systems*, 18(9):2435–2442, 2017.

[4] Goodness Oluchi Anyanwu, Cosmas Ifeanyi Nwakanma, Jae-Min Lee, and Dong-Seong Kim. Optimization of RBF-SVM kernel using grid search algorithm for DDoS attack detection in SDN-based VANET. *IEEE Internet of Things Journal*, 10(10):8477–8490, 2023.

[5] Yoshinori Aono, Takuya Hayashi, Le Trieu Phong, and Lihua Wang. Efficient homomorphic encryption with key rotation and security update. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 101(1):39–50, 2018.

[6] Giuseppe Ateniese, Daniel H Chou, Breno De Medeiros, and Gene Tsudik. Sanitizable signatures. In *Computer Security–ESORICS 2005: 10th European Symposium on Research in Computer Security, Milan, Italy, September 12-14, 2005. Proceedings 10*, pages 159–177. Springer, 2005.

[7] Eugene Bagdasaryan, Omid Poursaeed, and Vitaly Shmatikov. Differential privacy has disparate impact on model accuracy. *Advances in Neural Information Processing Systems*, 32, 2019.

[8] Wentao Bao, Qi Yu, and Yu Kong. Uncertainty-based traffic accident anticipation with spatio-temporal relational learning. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2682–2690, 2020.

[9] Paulo SLM Barreto, Marcos A Simplicio, Jefferson E Ricardini, and Harsh Kupwade Patil. Schnorr-based implicit certification: Improving the security and efficiency of vehicular communications. *IEEE Transactions on Computers*, 70(3):393–399, 2020.

[10] Beijing Municipal Ecological and Environmental Monitoring Center. Beijing air quality. [Online]. Available from: `http://zx.bjmemc.com.cn/`.

[11] Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In *Proceedings of the 13th SIGSAC Conference on Computer and Communications Security*, pages 390–399, 2006.

[12] Laura Bieker, Daniel Krajzewicz, AntonioPio Morra, Carlo Michelacci, and Fabio Cartolano. Traffic simulation for all: A real world traffic scenario from the city of bologna. In *Modeling Mobility with Open Data: 2nd SUMO Conference, Berlin, Germany*, pages 47–60. Springer, 2015.

[13] Norbert Bißmeyer, Hagen Stübing, Elmar Schoch, Stefan Götz, Jan Peter Stotz, and Brigitte Lonc. A generic public key infrastructure for securing Car-to-X communication. In *18th ITS World Congress, Orlando, USA*, volume 14, 2011.

[14] Sebastian Blasco, Javier Bustos-Jimenez, Giselle Font, Alejandro Hevia, and Marfa Grazia Prato. A three-layer approach for protecting smart-citizens privacy in crowdsensing projects. In *2015 34th International Conference of the Chilean Computer Science Society*, pages 1–5. IEEE, 2015.

[15] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

[16] Dan Boneh, Craig Gentry, Shai Halevi, Frank Wang, and David J Wu. Private database queries using somewhat homomorphic encryption. In *International Conference on Applied Cryptography and Network Security*, pages 102–118. Springer, 2013.

[17] Angèle Bossuat and Xavier Bultel. Unlinkable and invisible γ-sanitizable signatures. In *International Conference on Applied Cryptography and Network Security*, pages 251–283. Springer, 2021.

[18] Luca Bravi, Luca Kubin, Stefano Caprasecca, Douglas Coimbra de Andrade, Matteo Simoncini, Leonardo Taccari, and Francesco Sambo. Detection of stop sign violations from dashcam data. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):5411–5420, 2021.

[19] Benedikt Brecht, Dean Therriault, André Weimerskirch, William Whyte, Virendra Kumar, Thorsten Hehn, and Roy Goudy. A security credential management system for V2X communications. *IEEE Transactions on Intelligent Transportation Systems*, 19(12):3850–3871, 2018.

[20] Christina Brzuska, Marc Fischlin, Anja Lehmann, and Dominique Schröder. Unlinkability of sanitizable signatures. In *Public Key Cryptography–PKC 2010: 13th International Conference on Practice and Theory in Public Key Cryptography, Paris, France, May 26-28, 2010. Proceedings 13*, pages 444–461. Springer, 2010.

[21] Xavier Bultel, Pascal Lafourcade, Russell WF Lai, Giulio Malavolta, Dominique Schröder, and Sri Aravinda Krishnan Thyagarajan. Efficient invisible and unlinkable sanitizable signatures. In *Public-Key Cryptography–PKC 2019: 22nd IACR International Conference on Practice and Theory of Public-Key Cryptography, Beijing, China, April 14-17, 2019, Proceedings, Part I 22*, pages 159–189. Springer, 2019.

[22] Sébastien Canard and Amandine Jambert. On extended sanitizable signature schemes. In *Cryptographers' Track at the RSA Conference*, pages 179–194. Springer, 2010.

[23] Antonio Celesti, Antonino Galletta, Lorenzo Carnevale, Maria Fazio, Aime Ĺay-Ekuakille, and Massimo Villari. An IoT cloud system for traffic monitoring and

vehicular accidents prevention based on mobile sensor data processing. *IEEE Sensors Journal*, 18(12):4795–4802, 2017.

[24] Jung Hee Cheon, Jean-Sébastien Coron, Jinsu Kim, Moon Sung Lee, Tancrede Lepoint, Mehdi Tibouchi, and Aaram Yun. Batch fully homomorphic encryption over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 315–335. Springer, 2013.

[25] Jae Gyeong Choi, Chan Woo Kong, Gyeongho Kim, and Sunghoon Lim. Car crash detection using ensemble deep learning and multimodal data from dashboard cameras. *Expert Systems with Applications*, 183:115400, 2021.

[26] Lemei Da, Yujue Wang, Yong Ding, Bo Qin, Xiaochun Zhou, Hai Liang, and Huiyong Wang. Cloud-assisted road condition monitoring with privacy protection in VANETs. In *2022 18th International Conference on Mobility, Sensing and Networking (MSN)*, pages 304–311. IEEE, 2022.

[27] Nejdet Dogru and Abdulhamit Subasi. Traffic accident detection using random forest classifier. In *2018 15th Learning and Technology Conference (L&T)*, pages 40–45. IEEE, 2018.

[28] Yang Du, Yu-E Sun, He Huang, Liusheng Huang, Hongli Xu, Yu Bao, and Hansong Guo. Bayesian co-clustering truth discovery for mobile crowd sensing systems. *IEEE Transactions on Industrial Informatics*, 16(2):1045–1057, 2019.

[29] Secil Ercan, Marwane Ayaida, and Nadhir Messai. An enhanced pseudonym certificates distribution mechanism for connected vehicles. *International Journal of Communication Systems*, 35(7):e5100, 2022.

[30] ETSI. Intelligent transport systems (ITS); security; pre-standardization study on pseudonym change management. *Technical Report ETSI TR 103 415 V1. 1.1 (2018-04)*, 2018.

[31] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Conference on the theory and application of cryptographic techniques*, pages 186–194. Springer, 1986.

[32] Nils Fleischhacker, Johannes Krupp, Giulio Malavolta, Jonas Schneider, Dominique Schröder, and Mark Simkin. Efficient unlinkable sanitizable signatures

from signatures with re-randomizable keys. In *Public-Key Cryptography–PKC 2016: 19th IACR International Conference on Practice and Theory in Public-Key Cryptography, Taipei, Taiwan, March 6-9, 2016, Proceedings, Part I*, pages 301–330. Springer, 2016.

[33] Honghao Gao, Can Liu, Youhuizi Li, and Xiaoxian Yang. V2VR: Reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability. *IEEE Transactions on Intelligent Transportation Systems*, 22(6):3533–3546, 2020.

[34] Jiechao Gao, Gunasekaran Manogaran, Tu N Nguyen, Seifedine Kadry, Ching-Hsien Hsu, and Priyan Malarvizhi Kumar. A vehicle-consensus information exchange scheme for traffic management in vehicular ad-hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):19602–19612, 2022.

[35] Garima Gupta, Ritwik Singh, Ashish Singh Patel, and Muneendra Ojha. Accident detection using time-distributed model in videos. In *Proceedings of Fifth International Congress on Information and Communication Technology*, pages 214–223. Springer, 2021.

[36] Maanak Gupta, James Benson, Farhan Patwa, and Ravi Sandhu. Secure V2V and V2I communication in intelligent transportation using cloudlets. *IEEE Transactions on Services Computing*, 15(4):1912–1925, 2020.

[37] Khalid Abdel Hafeez, Lian Zhao, Zaiyi Liao, and Bobby Ngok-Wah Ma. A fuzzy-logic-based cluster head selection algorithm in VANETs. In *2012 IEEE International Conference on Communications (ICC)*, pages 203–207. IEEE, 2012.

[38] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in Neural Information Processing Systems*, 28, 2015.

[39] Xu Han, Daxin Tian, Jianshan Zhou, Xuting Duan, Zhengguo Sheng, and Victor CM Leung. Privacy-preserving proxy re-encryption with decentralized trust management for MEC-empowered VANETs. *IEEE Transactions on Intelligent Vehicles*, 8(8):4105–4119, 2023.

[40] Behnam Hassanabadi, Christine Shea, Le Zhang, and Shahrokh Valaee. Clustering in vehicular ad hoc networks using affinity propagation. *Ad Hoc Networks*, 13:535–548, 2014.

[41] Chunrong He, Guiyan Liu, Songtao Guo, and Yuanyuan Yang. Privacy-preserving and low-latency federated learning in edge computing. *IEEE Internet of Things Journal*, 9(20):20149–20159, 2022.

[42] Rui Hu, Yuanxiong Guo, Hongning Li, Qingqi Pei, and Yanmin Gong. Personalized federated learning with differential privacy. *IEEE Internet of Things Journal*, 7(10):9530–9539, 2020.

[43] Xumin Huang, Peichun Li, Rong Yu, Yuan Wu, Kan Xie, and Shengli Xie. FedParking: A federated learning based parking space estimation with parked vehicle assisted edge computing. *IEEE Transactions on Vehicular Technology*, 70(9):9355–9368, 2021.

[44] Zonghao Huang, Rui Hu, Yuanxiong Guo, Eric Chan-Tin, and Yanmin Gong. DP-ADMM: ADMM-based distributed learning with differential privacy. *IEEE Transactions on Information Forensics and Security*, 15:1002–1012, 2019.

[45] IBM Security Team. Specification of the Identity Mixer Cryptographic Library Version 2.3.0, 2010.

[46] IEEE Std 1609.2.1-2022 (Revision of IEEE Std 1609.2.1-2020). IEEE Standard for Wireless Access in Vehicular Environments (WAVE) - Certificate Management Interfaces for End Entities, 2022.

[47] Earnest Paul Ijjina, Dhananjai Chand, Savyasachi Gupta, and K Goutham. Computer vision-based accident detection in traffic surveillance. In *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–6. IEEE, 2019.

[48] Earnest Paul Ijjina and Sanjay Kumar Sharma. Accident detection from dashboard camera video. In *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–4. IEEE, 2019.

[49] Ahmed Imteaj and M Hadi Amini. FedPARL: Client activity and resource-oriented lightweight federated learning model for resource-constrained heterogeneous IoT environment. *Frontiers in Communications and Networks*, 2:657653, 2021.

[50] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M Hadi Amini. A survey on federated learning for resource-constrained IoT devices. *IEEE Internet of Things Journal*, 9(1):1–24, 2021.

[51] Yuang Jiang, Shiqiang Wang, Victor Valls, Bong Jun Ko, Weihan Lee, Kin K Leung, and Leandros Tassiulas. Model pruning enables efficient federated learning on edge devices. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):10374–10386, 2023.

[52] Haiming Jin, Lu Su, and Klara Nahrstedt. Theseus: Incentivizing truth discovery in mobile crowd sensing systems. In *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 1–10, 2017.

[53] Ammara Anjum Khan, Mehran Abolhasan, and Wei Ni. An evolutionary game theoretic approach for stable and optimized clustering in VANETs. *IEEE Transactions on Vehicular Technology*, 67(5):4501–4513, 2018.

[54] Qinglei Kong, Rongxing Lu, Maode Ma, and Haiyong Bao. A privacy-preserving sensory data sharing scheme in Internet of Vehicles. *Future Generation Computer Systems*, 92:644–655, 2019.

[55] Qi Li, Yaliang Li, Jing Gao, Bo Zhao, Wei Fan, and Jiawei Han. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, pages 1187–1198, 2014.

[56] Shancang Li, Shanshan Zhao, Geyong Min, Lianyong Qi, and Gang Liu. Lightweight privacy-preserving scheme using homomorphic encryption in industrial Internet of Things. *IEEE Internet of Things Journal*, 9(16):14542–14550, 2022.

[57] Shike Li, Jianbin Li, Jiaming Pei, Sixing Wu, Shen Wang, and Long Cheng. Eco-CSAS: A safe and eco-friendly speed advisory system for autonomous ve-

hicle platoon using consortium blockchain. *IEEE Transactions on Intelligent Transportation Systems*, 24(7):7802–7812, 2023.

[58] Ting Li, Shangsheng Xie, Zhiwen Zeng, Mianxiong Dong, and Anfeng Liu. Atps: An AI based trust-aware and privacy-preserving system for vehicle managements in sustainable VANETs. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):19837–19851, 2022.

[59] Zhuoqian Li, Shuo Yang, Fan Wu, Xiaofeng Gao, and Guihai Chen. Holmes: Tackling data sparsity for truth discovery in location-aware mobile crowdsensing. In *2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems*, pages 424–432. IEEE, 2018.

[60] Yangfan Liang, Entao Luo, and Yining Liu. Physically secure and conditional-privacy authenticated key agreement for VANETs. *IEEE Transactions on Vehicular Technology*, 72(6):7914–7925, 2023.

[61] Ping Liu, Xingfu Wang, Ammar Hawbani, Bei Hua, Liang Zhao, and Zhi Liu. Beta: Beacon-based traffic-aware routing in vehicular ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, 23(12):24206–24219, 2022.

[62] Rui Liu, Yun Lu, and Jianping Pan. Flexible non-interactive short-term implicit certificate generation for VANETs. *arXiv preprint arXiv:2402.02607*, 2024.

[63] Rui Liu and Jianping Pan. AirQ: A privacy-preserving truth discovery framework for vehicular air quality monitoring. In *2020 16th International Conference on Mobility, Sensing and Networking (MSN)*, pages 65–72. IEEE, 2020.

[64] Rui Liu and Jianping Pan. Lightweight privacy-preserving truth discovery for vehicular air quality monitoring. *Digital Communications and Networks*, 9(1):280–291, 2023.

[65] Rui Liu and Jianping Pan. CRS: A privacy-preserving two-layered distributed machine learning framework for IoV. *IEEE Internet of Things Journal*, 11(1):1080–1095, 2024.

[66] Rui Liu and Jianping Pan. Schnorr approval-based secure and privacy-preserving IoV data aggregation. *arXiv preprint arXiv:2402.09621*, 2024.

[67] Rongxing Lu, Xiaodong Lin, Haojin Zhu, and Xuemin Shen. SPARK: A new VANET-based smart parking scheme for large parking lots. In *IEEE INFOCOM 2009*, pages 1413–1421. IEEE, 2009.

[68] Xiaomin Ma and Kishor S Trivedi. SINR-based analysis of IEEE 802.11p/bd broadcast VANETs for safety services. *IEEE Transactions on Network and Service Management*, 18(3):2672–2686, 2021.

[69] Abbass Madi, Oana Stan, Aurélien Mayoue, Arnaud Grivet-Sébert, Cédric Gouy-Pailler, and Renaud Sirdey. A secure federated learning framework using homomorphic encryption and verifiable computing. In *2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS)*, pages 1–8. IEEE, 2021.

[70] Gregory Maxwell, Andrew Poelstra, Yannick Seurin, and Pieter Wuille. Simple Schnorr multi-signatures with applications to Bitcoin. *Designs, Codes and Cryptography*, 87(9):2139–2164, 2019.

[71] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.

[72] Chuishi Meng, Wenjun Jiang, Yaliang Li, Jing Gao, Lu Su, Hu Ding, and Yun Cheng. Truth discovery on crowd sensing of correlated entities. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pages 169–182, 2015.

[73] Chenglin Miao, Wenjun Jiang, Lu Su, Yaliang Li, Suxin Guo, Zhan Qin, Houping Xiao, Jing Gao, and Kui Ren. Cloud-enabled privacy-preserving truth discovery in crowd sensing systems. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pages 183–196, 2015.

[74] Chenglin Miao, Wenjun Jiang, Lu Su, Yaliang Li, Suxin Guo, Zhan Qin, Houping Xiao, Jing Gao, and Kui Ren. Privacy-preserving truth discovery in crowd sensing systems. *ACM Transactions on Sensor Networks*, 15(1):1–32, 2019.

[75] Ministry of Environmental Protection of the People's Republic of China. *Technical Regulation on Ambient Air Quality Index (on*

trial), 2012. https://www.mee.gov.cn/ywgz/fgbz/bz/bzwb/jcffbz/201203/W020120410332725219541.pdf.

[76] MIRACL. Multiprecision Integer and Rational Arithmetic Cryptographic Library. [Open source]. Available from: https://github.com/miracl/MIRACL, 2022.

[77] Shafika Showkat Moni and D Manivannan. A scalable and distributed architecture for secure and privacy-preserving authentication and message dissemination in VANETs. *Internet of Things*, 13:100350, 2021.

[78] Motional. Motional and Uber Eats launch autonomous deliveries in Santa Monica. [Online]. Available from: https://motional.com/news/motional-and-uber-eats-launch-autonomous-deliveries-santa-monica, 2022.

[79] Hamed Noori, David G Michelson, and Kevin Henry. Reporting spectrum misbehaviour using the IEEE 1609 security credential management system. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pages 1–6. IEEE, 2020.

[80] ns-3 community. ns-3: A discrete-event Network Simulator for Internet Systems. [Open source]. Available from: https://www.nsnam.org, 2022.

[81] Nsikak P Owoh and M Mahinderjit Singh. Security analysis of mobile crowd sensing applications. *Applied Computing and Informatics*, 18(1/2):2–21, 2022.

[82] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 223–238. Springer, 1999.

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

[84] Le Trieu Phong, Yoshinori Aono, Takuya Hayashi, Lihua Wang, and Shiho Moriai. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5):1333–1345, 2018.

[85] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13:361–396, 2000.

[86] Zuber Purahoo and Sudha Cheerkoot-Jalim. SenseAPP: An IoT-based mobile crowdsensing application for smart cities. In *2020 3rd International Conference on Emerging Trends in Electrical, Electronic and Communications Engineering (ELECOM)*, pages 47–52, 2020.

[87] Jiayu Qi, Tianhan Gao, Xinyang Deng, and Cong Zhao. A pseudonym-based certificateless privacy-preserving authentication scheme for VANETs. *Vehicular Communications*, 38:100535, 2022.

[88] Liping Qian, Yuan Wu, Haibo Zhou, and Xuemin Shen. Dynamic cell association for non-orthogonal multiple-access V2S networks. *IEEE Journal on Selected Areas in Communications*, 35(10):2342–2356, 2017.

[89] Khaled Rabieh, Mohamed MEA Mahmoud, and Mohamed Younis. Privacy-preserving route reporting schemes for traffic management systems. *IEEE Transactions on Vehicular Technology*, 66(3):2703–2713, 2016.

[90] Gunasekaran Raja, Sudha Anbalagan, Geetha Vijayaraghavan, Sudhakar Theerthagiri, Saran Vaitangarukav Suryanarayan, and Xin-Wen Wu. SP-CIDS: Secure and private collaborative IDS for VANETs. *IEEE Transactions on Intelligent Transportation Systems*, 22(7):4385–4393, 2021.

[91] Zaydoun Y Rawshdeh and Syed Masud Mahmud. Toward strongley connected clustering structure in vehicular ad hoc networks. In *2009 IEEE 70th Vehicular Technology Conference Fall*, pages 1–5. IEEE, 2009.

[92] Mengying Ren, Jun Zhang, Lyes Khoukhi, Houda Labiod, and Véronique Vèque. A unified framework of clustering approach in vehicular ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, 19(5):1401–1414, 2017.

[93] Yilong Ren, Han Jiang, Xiaoyuan Feng, Yanan Zhao, Runkun Liu, and Haiyang Yu. Acp-based modeling of the parallel vehicular crowd sensing system: Framework, components and an application example. *IEEE Transactions on Intelligent Vehicles*, 8(2):1536–1548, 2022.

[94] Eldar Šabanovič, Vidas Žuraulis, Olegas Prentkovskis, and Viktor Skrickij. Identification of road-surface type using deep neural networks for friction coefficient estimation. *Sensors*, 20(3):612, 2020.

[95] Arijet Sarker, SangHyun Byun, Wenjun Fan, and Sang-Yoon Chang. Blockchain-based root of trust management in security credential management system for vehicular communications. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, pages 223–231, 2021.

[96] Suhail Mohmad Shah and Vincent KN Lau. Model compression for communication efficient federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 34(9):5937–5951, 2023.

[97] Christine Shea, Behnam Hassanabadi, and Shahrokh Valaee. Mobility-based clustering in VANETs using affinity propagation. In *GLOBECOM 2009-2009 IEEE Global Telecommunications Conference*, pages 1–6. IEEE, 2009.

[98] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(11):2298–2304, 2016.

[99] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 2015 ACM SIGSAC Conference on Computer and Communications Security*, pages 1310–1321, 2015.

[100] Matteo Simoncini, Douglas Coimbra de Andrade, Leonardo Taccari, Samuele Salti, Luca Kubin, Fabio Schoen, and Francesco Sambo. Unsafe maneuver classification from dashcam video and GPS/IMU sensors using spatio-temporal attention selector. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):15605–15615, 2022.

[101] Rajarajan Sivaraj, Aravind Kota Gopalakrishna, M Girish Chandra, and P Balamuralidhar. QoS-enabled group communication in integrated VANET-LTE heterogeneous wireless networks. In *2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 17–24. IEEE, 2011.

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

[103] Jingcheng Song, Weizheng Wang, Thippa Reddy Gadekallu, Jianyu Cao, and Yining Liu. EPPDA: An efficient privacy-preserving data aggregation federated learning scheme. *IEEE Transactions on Network Science and Engineering*, 10(5):3047–3057, 2023.

[104] Leonardo Taccari, Francesco Sambo, Luca Bravi, Samuele Salti, Leonardo Sarti, Matteo Simoncini, and Alessandro Lori. Classification of crash and near-crash events from dashcam videos and telematics. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2460–2465. IEEE, 2018.

[105] Wenjuan Tang, Ju Ren, Kun Deng, and Yaoxue Zhang. Secure data aggregation of lightweight E-healthcare IoT devices with fair incentives. *IEEE Internet of Things Journal*, 6(5):8714–8726, 2019.

[106] Shrikant Tangade, Sunilkumar S Manvi, and Pascal Lorenz. Trust management scheme based on hybrid cryptography for secure communications in VANETs. *IEEE Transactions on Vehicular Technology*, 69(5):5232–5243, 2020.

[107] The University of Texas at Dallas Data Security and Privacy Lab. Paillier Threshold Encryption Toolbox. [Open source]. Available from: `http://www.cs.utdallas.edu/dspl/cgi-bin/pailliertoolbox`, 2010.

[108] Transportation Association of Canada. *Geometric design guide for Canadian roads*, 2017. [Online]. Available from: `https://www.tac-atc.ca/en/publications-and-resources/geometric-design-guide-canadian-roads`.

[109] Geoff Twardokus and Hanif Rahbari. Vehicle-to-nothing? Securing C-V2X against protocol-aware DoS attacks. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pages 1629–1638. IEEE, 2022.

[110] Seyhan Ucar, Sinem Coleri Ergen, and Oznur Ozkasap. Multihop-cluster-based IEEE 802.11p and LTE hybrid architecture for VANET safety message dissemination. *IEEE Transactions on Vehicular Technology*, 65(4):2621–2636, 2015.

[111] Ikram Ullah, Abdul Wahid, Munam Ali Shah, and Abdul Waheed. VBPC: Velocity based pseudonym changing strategy to protect location privacy of vehicles in VANET. In *2017 International Conference on Communication Technologies (ComTech)*, pages 132–137. IEEE, 2017.

[112] Biying Wang, Zheng Chang, Zhenyu Zhou, and Tapani Ristaniemi. Reliable and privacy-preserving task recomposition for crowdsensing in vehicular fog computing. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pages 1–6. IEEE, 2018.

[113] Youchiun Wang and Guanwei Chen. Efficient data gathering and estimation for metropolitan air quality monitoring by using vehicular sensor networks. *IEEE Transactions on Vehicular Technology*, 66(8):7234–7248, 2017.

[114] Zhihua Wang, Yu Liu, Jiarui Wang, Zeminghui Li, Zhenyu Li, Xiaolong Yang, Fazhi Qi, and Hongyong Jia. A reliable physical layer key generation scheme based on RSS and LSTM network in VANET. *IEEE Internet of Things Journal*, 11(1):692–707, 2024.

[115] Lu Wei, Jie Cui, Yan Xu, Jiujun Cheng, and Hong Zhong. Secure and lightweight conditional privacy-preserving authentication for securing traffic emergency messages in VANETs. *IEEE Transactions on Information Forensics and Security*, 16:1681–1695, 2021.

[116] Daozhen Xi, Haixia Zhang, Yi Cao, and Dongfeng Yuan. An RSUs-assisted hybrid emergency messages broadcasting protocol for VANETs. *IEEE Internet of Things Journal*, 10(19):17479–17489, 2023.

[117] Fengli Xu, Yong Li, Huandong Wang, Pengyu Zhang, and Depeng Jin. Understanding mobile traffic patterns of large scale cellular towers in urban environment. *IEEE/ACM Transactions on Networking*, 25(2):1147–1161, 2016.

[118] Guowen Xu, Hongwei Li, Sen Liu, Mi Wen, and Rongxing Lu. Efficient and privacy-preserving truth discovery in mobile crowd sensing systems. *IEEE Transactions on Vehicular Technology*, 68(4):3854–3865, 2019.

[119] Guowen Xu, Hongwei Li, Chen Tan, Dongxiao Liu, Yuanshun Dai, and Kan Yang. Achieving efficient and privacy-preserving truth discovery in crowd sensing systems. *Computers & Security*, 69:114–126, 2017.

[120] Jielong Yang, Junshan Wang, and Wee Peng Tay. Using social network information in community-based bayesian truth discovery. *IEEE Transactions on Signal and Information Processing over Networks*, 5(3):525–537, 2019.

[121] Ming Yang, Rongwang Jiang, Shuang Wei, and Caofang Long. Scalable and auditable self-agent pseudonym management scheme for intelligent transportation systems. *Computers and Electrical Engineering*, 108:108735, 2023.

[122] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

[123] Yafang Yang, Lei Zhang, Yunlei Zhao, Kim-Kwang Raymond Choo, and Yan Zhang. Privacy-preserving aggregation-authentication scheme for safety warning system in fog-cloud based VANET. *IEEE Transactions on Information Forensics and Security*, 17:317–331, 2022.

[124] Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan, and Yang Liu. BatchCrypt: Efficient homomorphic encryption for Cross-Silo federated learning. In *2020 USENIX Annual Technical Conference (USENIX ATC 20)*, pages 493–506, 2020.

[125] Daniel Yue Zhang, Rungang Han, Dong Wang, and Chao Huang. On robust truth discovery in sparse social media sensing. In *2016 IEEE International Conference on Big Data*, pages 1076–1081. IEEE, 2016.

[126] Degan Zhang, Hui Ge, Ting Zhang, Yu-Ya Cui, Xiaohuan Liu, and Guoqiang Mao. New multi-hop clustering algorithm for vehicular ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, 20(4):1517–1530, 2018.

[127] Hongyi Zhang, Jan Bosch, and Helena Holmström Olsson. End-to-end federated learning for autonomous driving vehicles. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2021.

[128] Hongyi Zhang, Jan Bosch, and Helena Holmström Olsson. Real-time end-to-end federated learning: An automotive case study. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 459–468. IEEE, 2021.

[129] Zhili Zhou, Akshat Gaurav, Brij Bhooshan Gupta, Miltiadis D Lytras, and Imran Razzak. A fine-grained access control and security approach for intelligent vehicular transport in 6G communication system. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):9726–9735, 2021.

Appendix A

Proof of Theorem 4.1

We follow the same strategy as the original proof of MuSig [70]. For brevity, we omit the same definitions and some details of the proof but highlight the difference from that of MuSig.

Proof. We first construct an algorithm $\mathbb{A}$ simulating the oracles, running the forger (CH), answering the queries, and returning a forgery unless some bad events happen. We assume there is a honest member vehicle v_{i^*} with the key pair (sk_{i^*}, pk_{i^*}). The input of $\mathbb{A}$ includes pk_{i^*}, and uniformly random strings $h_{0,1}, h_{0,2}, \ldots, h_{0,n_{\mathrm{q}}-1}$ and $h_{1,1}, h_{1,2}, \ldots, h_{1,n_{\mathrm{q}}}$ where $n_{\mathrm{q}} = n_{\mathrm{qs}} + n_{\mathrm{qh}} + 1$.

Query A.1 Approval query $(L_{\mathrm{pk}}, c_{i^*}, info_{i^*})$

1: **if** $pk_{i^*} \notin L_{\mathrm{pk}}$ **then return** $\perp$. **end if**
2: Supposes $L_{\mathrm{pk}} = \{pk_{i^*}, pk_2, \ldots, pk_{n_{\mathrm{v}}}\}$, i.e., $i^* = 1$;
3: **if** $T_{\mathrm{agg}}(L_{\mathrm{pk}} \| pk_1)$ is undefined **then**
4: Internal query: hash query $\mathrm{H}_{\mathrm{agg}}(L_{\mathrm{pk}} \| pk_1)$;
5: **end if**
6: Sets $a_i^3 = T_{\mathrm{agg}}(L_{\mathrm{pk}} \| pk_i)$ for each i; $\widetilde{pk} = \prod_{i=1}^{n_{\mathrm{v}}} pk_i^{a_i^3}$;
7: Sets $ctr_1 ++$ and sets $e = h_{1,ctr_1}$;
8: Generates a random value as the signature $s_1 \xleftarrow{\$} \mathbb{Z}_q$;
9: Calculates R_1 from s_1: $R_1 = g^{s_1 (pk_1)^{-a_1^3 e}}$;
10: **if** $T_{\mathrm{com}}(R_1 \| c_1 \| info_1)$ is defined **then**
11: Sets the flag $BadCom_1$ to true; **return** $\perp$.
12: **end if**
13: Internal query: hash query $\mathrm{H}_{\mathrm{com}}(R_1 \| c_1 \| info_1)$;
14: Sets $com_1 = T_{\mathrm{com}}(R_1 \| c_1 \| info_1)$ and sends it to CH;

15: Receives $L_{\mathrm{com}} = \{com_i \mid i \in N_{\mathrm{v}}\}$ collected by CH;
16: For $\forall i \in N_{\mathrm{v}} \wedge i \neq 1$, looks for entries $(R_i \| c_i \| info_i)$ s.t. $T_{\mathrm{com}}(R_i \| c_i \| info_i) = com_i$;
17: **if** For some i, more than one entry can be found **then**
18: Sets the flag $BadCom_2$ to true; **return** $\perp$.
19: **end if**
20: **if** For some i, no such value can be found **then**
21: Sends $m_1 = \{R_1 \| c_1 \| info_1\}$ to CH;
22: Receives $m_i = \{R_i \| c_i \| info_i\}$ for $\forall i \in N_{\mathrm{v}} \wedge i \neq 1$;
23: Internal query: hash query $\mathrm{H}_{\mathrm{com}}(R_i \| c_i \| info_i)$ for these i;
24: **if** For some i, $\mathrm{H}_{\mathrm{com}}(R_i \| c_i \| info_i) \neq com_i$ **then**
25: Aborts the protocol.
26: **else**
27: Sets the flag $BadCom_3$ to true; **return** $\perp$.
28: **end if**
29: **end if**
30: **if** Exactly one entry can be found for each i **then**
31: Sends $m_1 = \{R_1 \| c_1 \| info_1\}$ to CH;
32: Receives $m_i = \{R_i \| c_i \| info_i\}$ for $\forall i \in N_{\mathrm{v}} \wedge i \neq 1$;
33: **if** For some i, $\mathrm{H}_{\mathrm{com}}(R_i \| c_i \| info_i) \neq com_i$ **then**
34: Aborts the protocol.
35: **end if**
36: Computes $\widetilde{R} = \prod_{i=1}^{n_{\mathrm{v}}} R_i$;
37: Computes the average value $\overline{data} = \frac{1}{n_{\mathrm{v}}} \sum_{i=1}^{n_{\mathrm{v}}} c_i$;
38: **if** $T_{\mathrm{app}}(\widetilde{pk} \| \widetilde{R} \| \overline{data})$ has already been defined **then**
39: Sets $BadProg$ to true; **return** $\perp$.
40: **else**
41: Assigns $T_{\mathrm{app}}(\widetilde{pk} \| \widetilde{R} \| \overline{data}) = e$;
42: **end if**
43: Sends s_1 to the forger, i.e., CH.
44: **end if**

Algorithm $\mathbb{A}$ is responsible for answering four kind of queries to CH: 1) hash query $\mathrm{H}_{\mathrm{com}}(R_i \| c_i \| info_i)$ where $info_i$ denotes $h_i \| \{\mathrm{E}_{key_{\mathrm{v}_{i,j}}}(f_i(j)) \mid j \in N_{\mathrm{v}} \wedge j \neq i\}$. $\mathbb{A}$ randomly assigns a value to the corresponding hash table entry, $T_{\mathrm{com}}(R_i \| c_i \| info_i) \xleftarrow{\$} \{0,1\}^{\mathrm{l_{hash}}}$, if it is undefined, and returns $T_{\mathrm{com}}(R_i \| c_i \| info_i)$. 2) Hash query $\mathrm{H}_{\mathrm{agg}}(L_{\mathrm{pk}} \| pk_i)$. 3) Hash

query $\mathrm{H}_{\mathrm{app}}(\widetilde{pk}\|\widetilde{R}\|\overline{data})$. These two queries remain the same as that in MuSig [70]. Corresponding hash tables are assigned randomly or directly from the input strings. Corresponding counters, ctr_0 and ctr_1, are increased. The queried hash values are returned as well. 4) Approval query, which simulates the sub-approval generation oracle as the honest cluster member v_{i^*}. We describe it in Query A.1 and highlight the difference from the signature query in MuSig as follows:

- In MuSig, the message that requires signing is given explicitly as input. However, in SADA, the average value is calculated by every vehicle in the cluster independently.
- We must exchange m_i first, calculate $\overline{data}$ (Steps 31–37), and then check whether $T_{\mathrm{app}}(\widetilde{pk}\|\widetilde{R}\|\overline{data})$ is already defined (Steps 38–42).
- Because of the different logic in Query A.1, there is no need to remain the flag *Alert* which is used along with the flag $BadCom_3$ in MuSig.

If the forger returns a forgery $(\widetilde{R}, \widetilde{s})$, $\mathbb{A}$ checks the validity of the forgery as described in the proof of MuSig [70]. Two bad events, *BadOrder* and *KeyColl*, may occur during this process. The major differences are as follows:

- Besides the corresponding L_{pk} of the forgery $(\widetilde{R}, \widetilde{s})$, the aggregated key $\widetilde{pk}$ is returned by the forger as well. We assume that the returned $\widetilde{pk}$ corresponds the set L_{pk} where $pk_{i^*} \in L_{\mathrm{pk}}$. This assumption is reasonable because we are analyzing the situation where the bad *CH* tries to generate a forgery on behalf of the whole cluster. In other words, it tries to upload a fake approval with the real aggregated public key.
- To return $\widetilde{pk}$, the forger must make a direct or internal $\mathrm{H}_{\mathrm{agg}}$ query. Besides, there is no need for $\mathbb{A}$ to calculate $\widetilde{pk}$ and check it. Thus, there is no need to check whether $T_{\mathrm{agg}}(L_{pk}\|pk_{i^*})$ is defined. This modification leads to the different number of $\mathrm{H}_{\mathrm{agg}}$ queries in $\mathbb{A}$.

Now we prove the lower bound of the accepting probability of $\mathbb{A}$. Suppose *CH* is a $(T_{\mathrm{fg}}, n_{\mathrm{qs}}, n_{\mathrm{qh}}, n_{\mathrm{v}}, \epsilon)$ forger, then the probability that $\mathbb{A}$ returns a forgery is as follows:

$$acc(\mathbb{A}) \geq \epsilon - \Pr[\mathrm{Bad}] \tag{A1}$$

where Pr[Bad] is the probability that the bad events happen. $BadCom_1$ happens when $T_{\mathrm{com}}(R_1\|c_1\|info_1)$ has been defined. $BadCom_2$ happens when two entries in T_{com} have

the same value. $BadCom_3$ happens when some $\mathrm{H}_{\mathrm{com}}(R_i\|c_i\|info_i)$ is undefined and gets set by chance to the value com_i. The probabilities remain the same as that in [70]:

$$\Pr[BadCom_1] \leq n_{\mathrm{qs}}(n_{\mathrm{v}}n_{\mathrm{qs}} + n_{\mathrm{qh}})/2^{l_{\mathrm{q}}-1}, \tag{A2}$$

$$\Pr[BadCom_2] \leq (n_{\mathrm{v}}n_{\mathrm{qs}} + n_{\mathrm{qh}})^2/2^{l_{\mathrm{hash}}+1}, \tag{A3}$$

$$\Pr[BadCom_3] \leq n_{\mathrm{v}}n_{\mathrm{qs}}/2^{l_{\mathrm{hash}}}. \tag{A4}$$

BadProg happens when $T_{\mathrm{app}}(\widetilde{pk}\|\widetilde{R}\|\overline{data})$ has already been defined. Every time the forger makes an approval query, it uses different $data_i$ (i.e., c_i) where $i \neq i^*$. We do not have any knowledge of how the forger chooses it so that we consider c_i is chosen uniformly at random. c_i has p_{mk} possible values. R_i has $2^{l_{\mathrm{q}}-1}$ possible values. There are n_{q} assignments to the table in total. Thus, for each approval query, the probability that $T_{\mathrm{app}}(\widetilde{pk}\|\widetilde{R}\|\overline{data})$ has already been defined is less than $n_{\mathrm{q}}/(p_{\mathrm{mk}}2^{l_{\mathrm{q}}-1})$. Considering that there are n_{qs} approval queries in total, the probability of *BadProg* is:

$$\Pr[BadProg] \leq n_{\mathrm{qs}}n_{\mathrm{q}}/(p_{\mathrm{mk}}2^{l_{\mathrm{q}}-1}). \tag{A5}$$

BadOrder happens when the assignment of $T_{\mathrm{agg}}(L_{pk}\|pk_{i^*})$ occurs after that of $T_{\mathrm{app}}(\widetilde{pk}\|\widetilde{R}\|\overline{data})$. Different with that in MuSig, there are at most $n_{\mathrm{qs}} + n_{\mathrm{qh}}$ assignments of $\mathrm{H}_{\mathrm{agg}}(L_{\mathrm{pk}}\|pk_i)$. The probability of *BadOrder* is:

$$\Pr[BadOrder] \leq (n_{\mathrm{qs}} + n_{\mathrm{qh}})n_{\mathrm{q}}/2^{l_{\mathrm{hash}}}. \tag{A6}$$

Similarly, the probability of *KeyColl*, which happens when two sets of public keys have the same aggregated key, is changed:

$$\Pr[KeyColl] \leq (n_{\mathrm{qs}} + n_{\mathrm{qh}})^2/2^{l_{\mathrm{hash}}}. \tag{A7}$$

With (A1), (A2), (A3), (A4), (A5), (A6) and (A7), we give a lower bound for $acc(\mathbb{A})$ as follows:

$$acc(\mathbb{A}) \geq \epsilon - 2(\frac{p_{\mathrm{mk}} + 1}{p_{\mathrm{mk}}})\frac{n_{\mathrm{qs}}(n_{\mathrm{v}}n_{\mathrm{qs}} + n_{\mathrm{qh}})}{2^{l_{\mathrm{q}}}} - \frac{4(n_{\mathrm{v}}n_{\mathrm{qs}} + n_{\mathrm{qh}})^2}{2^{l_{\mathrm{hash}}}}. \tag{A8}$$

Now we can construct an algorithm $\mathbb{B}$, which runs $\mathbb{A}$ twice and forks the execution of the forger on the answer to the query $\mathrm{H}_{\mathrm{app}}(\widetilde{pk}\|\widetilde{R}\|\overline{data})$. With $\mathbb{B}$, the discrete logarithm of $\widetilde{pk}$ can be retrieved. According to the generalized forking lemma [11],

the accepting probability of $\mathbb{B}$ can be calculated as follows:

$$\begin{aligned} acc(\mathbb{B}) \geq & acc(\mathbb{A}) \cdot (\frac{acc(\mathbb{A})}{n_{\mathrm{q}}} - \frac{1}{2^{l_{\mathrm{hash}}}}) \\ \geq & \frac{\epsilon^2}{n_{\mathrm{qs}} + n_{\mathrm{qh}} + 1} - 4(\frac{p_{\mathrm{mk}} + 1}{p_{\mathrm{mk}}}) \frac{n_{\mathrm{qs}}(n_{\mathrm{v}} n_{\mathrm{qs}} + n_{\mathrm{qh}})}{2^{l_{\mathrm{q}}}} \\ & - \frac{8(n_{\mathrm{v}} n_{\mathrm{qs}} + n_{\mathrm{qh}})^2 + 1}{2^{l_{\mathrm{hash}}}}. \end{aligned} \tag{A9}$$

Algorithm $\mathbb{C}$ runs $\mathbb{B}$ twice and forks the execution on the answer to $\mathrm{H}_{\mathrm{agg}}(L_{\mathrm{pk}} \| pk_i)$. With $\mathbb{C}$, the discrete logarithm of pk_{i^*} can be retrieved. We omit the details of the double-forking lemma [70] for brevity. The accepting probability of $\mathbb{C}$ is bounded as follows:

$$\begin{aligned} acc(\mathbb{C}) \geq & acc(\mathbb{B}) \cdot (\frac{acc(\mathbb{B})}{n_{\mathrm{q}} - 1} - \frac{1}{2^{l_{\mathrm{hash}}}}) \\ \geq & \frac{\epsilon^4}{(n_{\mathrm{qs}} + n_{\mathrm{qh}} + 1)^2 (n_{\mathrm{qs}} + n_{\mathrm{qh}})} - 8(\frac{p_{\mathrm{mk}} + 1}{p_{\mathrm{mk}}}) \\ & \frac{n_{\mathrm{qs}}(n_{\mathrm{v}} n_{\mathrm{qs}} + n_{\mathrm{qh}})}{2^{l_{\mathrm{q}}}} - \frac{16(n_{\mathrm{v}} n_{\mathrm{qs}} + n_{\mathrm{qh}})^2 + 3}{2^{l_{\mathrm{hash}}}}. \end{aligned} \tag{A10}$$

Compared with that in MuSig proof, the running time of $\mathbb{C}$ changes a bit. The size of the table T_{agg} is at most $n_{\mathrm{v}}(n_{\mathrm{qs}} + n_{\mathrm{qh}})$ in $\mathbb{A}$ while others remain the same. Suppose the time of each table assignment is $\mathcal{O}(1)$. We use $T_{\mathrm{G\_MUL}}$ to denote the point multiplication on $\mathbb{G}$. The running time of $\mathbb{C}$, $T_{\mathbb{C}}$, is as follows:

$$T_{\mathbb{C}} = 4T_{\mathrm{fg}} + 4\mathcal{O}(n_{\mathrm{v}}(n_{\mathrm{qs}} + n_{\mathrm{qh}})) + 4n_{\mathrm{v}} T_{\mathrm{G\_MUL}}. \tag{A11}$$

■