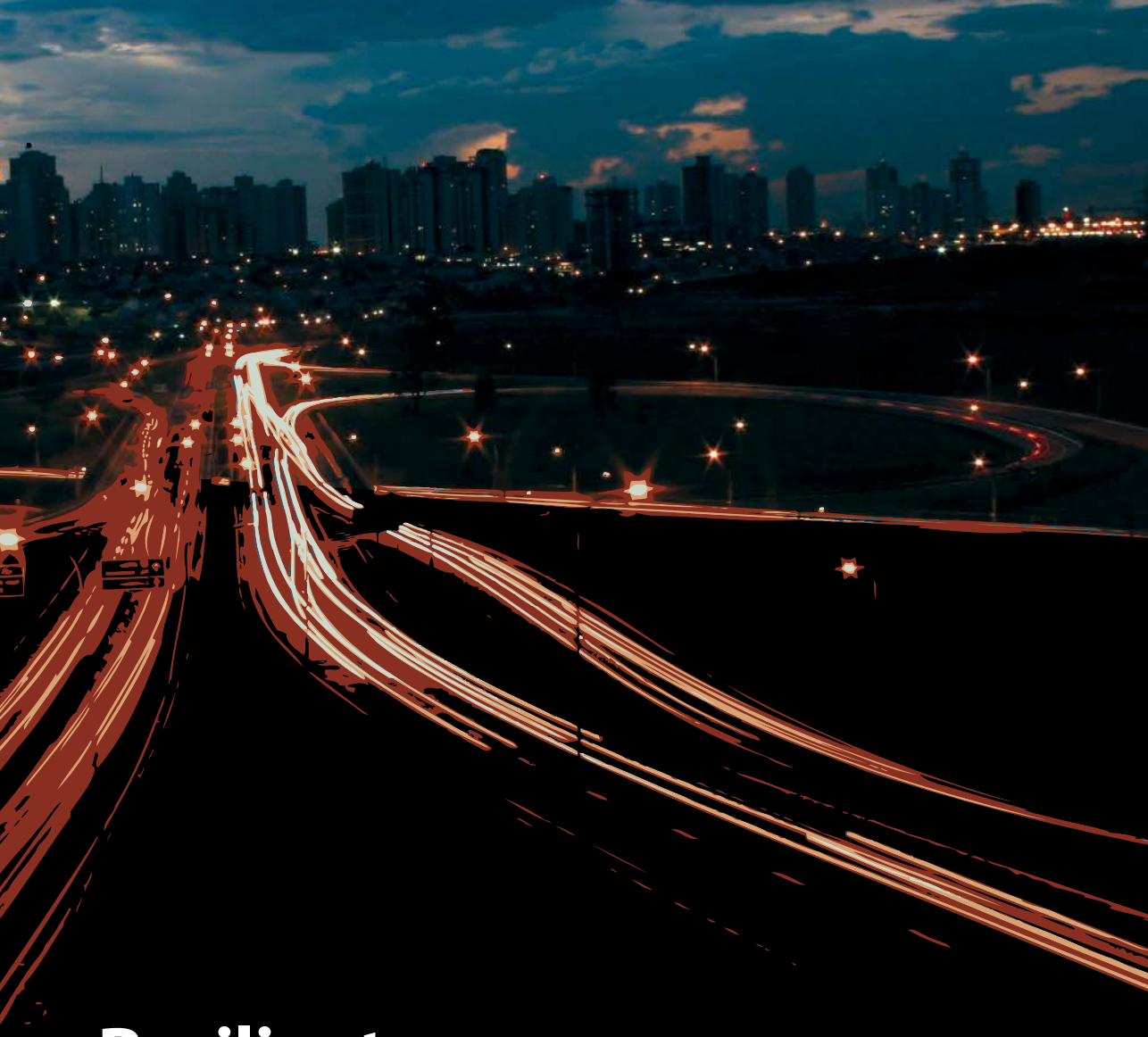


Resilient In-Network Aggregation for Vehicular Networks

Stefan Dietzel

RESILIENT IN-NETWORK AGGREGATION FOR VEHICULAR NETWORKS

STEFAN DIETZEL

SAMENSTELLING PROMOTIECOMMISSIE

prof. dr.	P. M. G. Apers	Universiteit Twente
prof. dr.	F. E. Kargl	Universiteit Twente, Universität Ulm
prof. dr.	P. H. Hartel	Universiteit Twente
prof. dr. ir.	A. Pras	Universiteit Twente
prof. dr.	B. Scheuermann	Humboldt Universität zu Berlin
prof. dr.-ing.	F. J. Hauck	Universität Ulm
dr.	G. J. Heijenk	Universiteit Twente
dr.	E. Schoch	Audi AG

CTIT Ph.D. thesis series No. 14-336
Centre for Telematics and Information Technology
P.O. Box 217, 7500 AE Enschede, The Netherlands

IPA Dissertation Series No. 2015-07
The research in this thesis has been carried out under the auspices of the research school IPA (Institute for Programming research and Algorithmics).

ISBN: 978-90-365-3852-7
ISSN: 1381-3617
DOI: 10.3990/1.9789036538527

Printed by: Ipskamp Drukkers B.V.
Typsetting: XeLaTeX

RESILIENT IN-NETWORK AGGREGATION FOR VEHICULAR NETWORKS

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. H. Brinksma,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op vrijdag 24 april 2015 om 14.45 uur

door

STEFAN DIETZEL

geboren op 14 februari 1983
te Bad Wildungen, Duitsland

Dit proefschrift is goedgekeurd door:

prof. dr. F. E. Kargl
dr. G. J. Heijenk

ABSTRACT

Applications for vehicular ad hoc networks (VANETs) are an active field of research with the potential to significantly contribute to driver safety, traffic efficiency, and comfort. Messages are typically exchanged and forwarded between vehicles using wireless communication, thereby creating a wireless ad hoc network. Especially traffic efficiency applications require the dissemination of information over long distances. For instance, vehicles need to be informed about traffic jams early enough to consider alternative navigation decisions. Each vehicle acts as creator and as forwarder of information to implement the required multihop information dissemination. Two of the most prevalent challenges in designing suitable ad hoc communication protocols are dealing with the *limited wireless channel capacity,* as well as ensuring the *resilience of communication protocols* against potential attackers.

The focus of this thesis is on the resilience of in-network information aggregation mechanisms for VANETs. In aggregation mechanisms, vehicles collaboratively exchange information and summarize this information as it is disseminated within the network. In contrast to traditional protocols, which often aggregate information at a centralized entity, the aggregation close to the information sources saves bandwidth and provides scalability. Yet, malicious users may be able to inject false information or even alter information summaries to disturb normal system operation. Both types of attacks are hard to detect, because original observations are usually discarded after aggregation and are not available to verify the correctness of claimed aggregated information. By addressing resilient in-network aggregation, this thesis provides solutions that *contribute to both channel capacity conservation and protocol resilience.*

The main contributions of this thesis are (a) a model of the in-network aggregation dissemination process; (b) a detailed security analysis of in-network aggregation mechanisms including the introduction of a taxonomy for security paradigms; (c) the design of four novel security mechanisms for in-network aggregation and (d) their detailed analysis and evaluation using network simulations; and (e) a framework that combines and adapts secure aggregation mechanisms based on situational context, as well as on attack likelihood derived from information exchange.

The model for in-network aggregation is comprised of an architecture model and an information flow model. It provides the foundation for understanding which components are essential in the design of aggregation mechanisms and for understanding how information spreads and evolves within the network.

The taxonomy of security paradigms, which is based on the modeling results, identifies use of cryptographic tools, interaction between vehicles to facilitate collaborative agreement, and data-consistency checks as most suitable security paradigms to provide resilience for in-network aggregation mechanisms.

Two security mechanisms that are based on cryptographic tools are proposed that are applicable to flexible, dynamic aggregation mechanisms. In contrast to related work, the proposed mechanisms do not rely on fixed road segments for aggregation, neither are they limited to the aggregation of binary events, such as presence of a traffic jam. Rather, they allow for flexible division of roads according to the velocities of the surrounding vehicles and are able to protect the integrity of more complex information, such as sets of average velocities that describe the current traffic situation.

The third mechanism, a cluster-based resilience mechanism, complements the first two mechanism proposals. By treating clusters as trustworthy units and implementing an efficient inter-cluster proof protocol, the clustering approach is especially applicable in dense traffic situations where the first two mechanisms may consume too much bandwidth.

The fourth mechanism, which focuses on data-consistency checks, provides protection that is orthogonal to the first three mechanism proposals. The mechanism leverages communication redundancy, which allows to detect inconsistencies between multiple redundant reports about the same event with less overhead than cryptography-based mechanisms.

Evaluation results of each mechanism indicate an inherent trade-off between bandwidth conservation and resilience against attackers. Therefore, a generic mechanism combination and adaptation framework is proposed that enables or disables mechanisms based on current traffic situation and to adapt mechanisms based on current attack likelihood. All necessary metrics, that is, traffic situation characterization and attack likelihood, are derived from the resilient aggregation mechanisms' exchanged information without requiring additional communication.

The traffic-dependent combination of mechanisms uses each mechanism in the situations for which it is most suitable while avoiding drawbacks of individual mechanisms in other traffic situations. Adaptation based on attack likelihood allows dynamic bandwidth-conserving configuration of mechanism parameters. When mechanisms find indications for attacks, they can be configured to use more bandwidth in order to increase resilience and detection accuracy. Likewise, the adaptation mechanism reduces bandwidth use when attacks are less likely.

The mechanism combination and adaptation framework demonstrates that bandwidth-efficient and scalable information dissemination using in-network aggregation is feasible while maintaining resilience against a broad range of possible attacks.

SAMENVATTING

Applicaties voor *VANETs* zijn een actief onderzoeksbereik met het potentiaal om de veiligheid, efficiëntie en comfort van bestuurders en verkeer sterk te verbeteren. In deze netwerken worden berichten uitgewisseld en doorgestuurd tussen voertuigen door middel van draadloze communicatie, waardoor een draadloos *ad-hoc* netwerk ontstaat. Vooral voor applicaties voor de verhoging van verkeersefficiëntie is het verspreiden van informatie over lange afstanden zeer belangrijk. Voortuigen moeten bijvoorbeeld op tijd geïnformeerd werden over verkeersopstoppingen, zodat men nog een alternatieve route kan kiezen. Elk voertuig dient als de maker en *forwarder* van informatie, waardoor de nodige *multi-hop* informatieverspreiding opgebouwd wordt. Twee van de belangrijkste uitdagingen van het ontwikkelen van passende *ad-hoc* communicatieprotocollen zijn de omgang met *beperkte capaciteit van het draadloze medium* en het garanderen van de *weerbaarheid van communicatieprotocollen* tegen mogelijke aanvallers.

De focus van dit proefschrift is de weerbaarheid van *in-network* aggregatiemechanismen voor *VANETs*. In aggregatiemechanismen wisselen voertuigen informatie uit en vatten deze regelmatig samen, terwijl het in het netwerk verdeeld wordt. In contract met traditionele protocollen, die meestal de informatie samenvatten bij een centrale entiteit, gebeurt de aggregatie hier dicht bij de informatiebronnen, zodat men bandbreedte kan besparen en daarmee de schaalbaarheid kan verhogen. Echter bestaat de mogelijkheid dat malafide gebruikers incorrecte informatie verspreiden, of zelfs aggregaten veranderen om het normale systeemgebruik te verstoren. Deze aanvallen zijn moeilijk te herkennen, omdat de originele observaties meestal worden verworpen nadat ze in een aggregaat opgenomen worden, en dus niet gebruikt kunnen worden om de correctheid van het aggregaat te verifiëren. Door de ontwikkeling van weerbare *in-network* aggregatie levert dit proefschrift oplossingen die *het gebruik van de capaciteit van het medium verbeteren, en de weerbaarheid van protocollen verhogen*.

De belangrijkste bijdragen van dit proefschrift zijn (a) een model van het *in-network* aggregatie en verspreidingsprocess; (b) een gedetaileerde veiligheidanalyse van *in-network* aggregatiemechanismen, inclusieve een taxonomie van beveiligingsparadigmen; (c) het ontwerpen van vier nieuwe beveiliginsmechanismen voor *in-network* aggregatie en (d) de gedetailleerde analyse en evaluatie door middel van gesimuleerde netwerken; en (e) een kader dat verschillende veilige aggregatiemechanismen combineert en aanpast op basis van de context en de waarschijnlijkheid van een aanval op basis van uitgewisselde informatie.

Het model voor *in-network* aggregatie bestaat uit een architectureel en een informatiestroom onderdeel. Het model geeft het fundament voor het begrip welke componenten voor het ontwerpen van aggregatiemechanismen essenti-

eel zijn en om duidelijk te maken hoe informatie zich in het netwerk ontwikkelt en verspreidt.

De taxonomie van beveiligingsparadigmen, die op basis van de resultaten van de modellering gegeven wordt, identificeert het gebruik van cryptografische mechanismen, de uitwisseling tussen voertuigen om overeenstemming te bereiken, en consistentiecontroles van de data als de meest gepaste beveiligingsparadigmen om de weerbaarheid van *in-network* aggregatiemechanismen te verhogen.

Twee beveiligingsmechanismen die op cryptografie baseren en van toepassing zijn op flexibele, dynamische aggregatiemechanismen worden voorgelegd. In tegenstelling tot de bestaande literatuur gaan deze mechanismen niet van vaste segmenten uit om de aggregaten te berekenen, noch zijn ze tot binaire gebeurtenissen, zoals het bestaan van een file, beperkt. De mechanismen maken het mogelijk om de weg flexibel in te delen op basis van bestaande verkeerssituaties en maken het mogelijk om de integriteit van complexe informatie te beschermen, zoals bijvoorbeeld de gemiddelde snelheiden van de voertuigen in de omgeving.

Het derde mechanisme baseert op *clustering* en ondersteunt de eerste twee mechanismen. Doordat clusters als vertrouwde eenheiden gezien worden, en door middel van een efficiënt bewijsprotocol tussen de clusters, is dit mechanisme vooral geschikt voor zeer dichte verkeerssituaties, waar de eerste twee mechanismen mogelijk te veel bandbreedte gebruiken.

Het vierde mechanisme gebruikt consistentiecontroles om orthogonale bescherming te leveren die met de eerste drie mechanismen kan worden gecombineerd. De mechanismen maken redundante communicatie mogelijk, waardoor men inconsistenties tussen verschillende redundante meldingen van dezelfde gebeurtenis met minder bandbreedte dan cryptografische mechanismen.

De evaluatie van de resultaten van de mechanismen wijzen op een inherente afweging tussen het besparen van bandbreedte en de weerbaarheid tegen aanvallers. Daarom wordt in dit proefschrift een algemeen combinatie en adaptatie kader opgebouwd dat het mogelijk maakt de juiste mechanismen dynamisch uit te kiezen op basis van de verkeerssitutatie, en om de resultaten aan te passen op basis van de waarschijnlijkheid van een aanval. Alle nodige metrieken, de kenmerken van de verkeerssituatie en de waarschijnlijkheid van een aanval, worden van de weerbare aggregatiemechanismen afgeleid en benodigen daardoor geen extra communicatie.

De verkeersafhankelijke combinatie van mechanismen gebruikt elk mechanisme in die situatie waar deze het meest geschikt is, waardoor de nadelen van de individuele mechanismen in andere verkeerssituaties gedeeltelijk opgeheven worden. De aanpassing op basis van aanvalswaarschijnlijkheid maakt het mogelijk om de mechanismen dynamisch in te stellen om bandbreedte te besparen. Zodra indicaties van mogelijke aanvallen gevonden worden kunnen de mechanismen zo aangepast worden dat ze meer bandbreedte gebruiken, maar daardoor aanvallen ook beter herkennen kunnen en het system weerbaarder

wordt. Dit maakt het ook mogelijk om bandbreedte te besparen in situaties waar aanvallen zeer onwaarschijnlijk zijn.

Het combinatie en adaptatie kader voor deze mechanismen maakt duidelijk dat het mogelijk is efficiënt met bandbreedte om te gaan, schaalbaar informatie te verspreiden en tegelijkertijd de weerbaarheid tegenover een breed scala aan aanvallen zeker te stellen door middel van *in-network* aggregatie.

CONTENTS

I

VEHICULAR AD HOC NETWORKS AND AGGREGATION

1 INTRODUCTION

1.1 MOTIVATION

Individual transport is a necessity for many of the world's citizens. The European automobile manufacturers association reports a total of 231 million private cars in use in the European Union in 2011 [204]. At the same time, over 30 thousand fatalities were reported in 2011. The European Commission (EC) has set forth guidelines to cut the number of fatalities in half by 2020 [237]. In these guidelines, the Commission specifically includes "use of modern technology to increase road safety," including "exchange [of] data and information between vehicles," which is often referred to as vehicular communication.

In contrast to existing technologies, which often observe the vehicle's local surroundings, vehicular communication has the potential to greatly enlarge the so-called telematic horizon. That is, drivers can be informed about farther away events, giving them more time to react accordingly. This potential is acknowledged in the European Commission's directive for the development of intelligent transportation systems [205]. The Commission highlights that "the application of information and communication technologies to the road transport sector […] will make a significant contribution to improving environmental performance, efficiency, including energy efficiency, safety and security of road transport." A strategic plan with similar objectives exists in the US [212]. Supported by legislation, vehicular communication is an active research area with continuing interest from the vehicular industry and researchers alike.

Inter-vehicle communication environment.

Inter-vehicle communication is implemented by creating a VANET, which is a mobile ad hoc network with unique communication requirements. The core idea of VANETs is to equip each car with wireless communication hardware, so-called dedicated short-range communication (DSRC) units to enable message exchange among vehicles [79]. A characterizing feature of VANETs is ephemeral communication, which is the result of high vehicle mobility. All vehicles periodically send out information about their current position, speed, and other parameters, and they are interested in receiving these information from other vehicles. Hence, all vehicles are both content producers and consumers. Moreover, each vehicle may act as forwarder to disseminate messages to farther away vehicles.

Vehicular networks can support a broad range of applications.

Application scenarios for VANETs can be broadly categorized into safety, efficiency, and entertainment applications [79]. Safety applications aim to reduce the number of traffic-related accidents. Their predominant communication pattern is a high-frequency exchange of information about other vehicles in the direct vicinity, which helps to improve situational awareness, for example, when changing lanes. In addition, specific warning messages can be used to inform approaching vehicles about potentially dangerous events, such as accidents or icy road stretches. Efficiency applications are envisioned to improve traffic flow

and, thereby, the driving experience. Some efficiency applications, like green light optimal speed advisory (GLOSA) [197], work on a local scope, like safety applications. But the majority of efficiency applications, such as dynamic route planners and global traffic flow optimization, depend on information gathered from large regions [223]. The entertainment category subsumes a number of applications that aim to provide Internet connectivity to cars to enable streaming and download of multimedia content and applications.

Message dissemination patterns.

A key differentiation factor of VANETs is that broadcast of information is the dissemination pattern of choice. Both in case of safety applications and in case of efficiency applications, the intended receivers of information are not *specific cars,* but the set of cars residing in *specific regions.* Target regions for safety events can often be localized to a small region adjacent to the event location. Geographic broadcast – often abbreviated as geocast – has been established as a suitable communication pattern for safety applications [202, 221]. For efficiency applications, information needs to be disseminated in larger regions [223]. As a simple example, consider a highway route planner. To recalculate routes in time, information about traffic jams ahead needs to be known at least several kilometers in advance. Ideally, the traffic situation on the surrounding part of the highway network would be known at a scale of dozens of kilometers.

Applications have a need for semantic summaries.

Consequently, each vehicle constantly needs to analyze a large set of information items. Here, analyzing means that vehicles use their raw information base to derive summarized, semantic objects and events. Traffic information systems combine reports from individual reports from vehicles to detect the location, extent, and severity of traffic jams and stretches of free flowing traffic; parking spot finders summarize the number of free spots in a parking lot; and so forth.

Data aggregation.

The basis for this kind of summarization is *data aggregation*, which encompasses all functions that take one or more input values and combine them to a single result that represents these input values. Aggregation can be as simple as calculating a sum or arithmetic mean and as complex as clustering a set of input values to derive their interrelation and structure. In traditional system designs, aggregation is often performed at a centralized point, which collects all information necessary for the foreseen summarization. This traditional approach is known as *destination* aggregation [20, 145]. Destination aggregation works as long as each data source has a – preferably direct – communication link with sufficient bandwidth to the destination.

Need for aggregation close to data sources.

In ad hoc networks in general and VANETs in particular, messages may need to be rebroadcasted by intermediate nodes to make all necessary information known to all interested parties. Such forwarding can easily overload the scarce wireless resources. Therefore, aggregation should be performed as close to the *source* as possible, leading to *in-network* aggregation. Instead of forwarding raw information unmodified, all forwarding vehicles potentially modify and summarize the information they receive. For example, vehicles that are part of the same traffic jam can immediately summarize their own and received atomic observations (e. g., "vehicle id x is standing still at location y and timestamp

z") and only rebroadcast the summarized information (e. g., "there is a traffic jam at location y, length l, timestamp z, containing n vehicles") to approaching traffic. Intuitively, aggregation performed close to the information sources reduces communication bandwidth usage, because summarized information can typically be represented more compactly than the original information items.

Necessity for resilient protocols.

To ensure utility of collected information, in-network aggregation protocols need to be resilient against malicious entities. Malicious entities are a threat to many VANET applications. In case of in-network aggregation, potential attackers may aim to simply disrupt normal system operation by injecting false information. Or they may manufacture specific traffic situations, such as purported traffic jams, to manipulate navigation decisions of other vehicles for personal gain. In some cases, attackers may even provoke dangerous driving maneuvers. For example, when they achieve that claimed traffic jams suddenly appear in navigation systems or real traffic jams suddenly disappear. All those attacks aim to convince other vehicles of a purported situation that deviates substantially from the real world situation. Resilient in-network aggregation protocols need to detect and filter such false information by protecting the integrity of information gathered by honest vehicles, as well as the integrity of aggregated summaries created by honest vehicles.

Threats arise from insider and outsider attackers.

Resilience must be ensured against outsider attackers, as well as insider attackers. Outsider attackers are entities that can participate in communication but are distinguishable from authorized entities [232]. For example, when authorized entities possess a cryptographic key pair that is certified by a trusted party, outsiders would not possess a certified key pair. The outsider messages can then be distinguished by examining signatures, public keys, and certificates [192]. Insider attackers are not distinguishable from honest vehicles in the same way. That is, insider attackers can use a certified key pair to create messages. It is conceivable that attackers of VANET applications will be insider attackers, which complicates protection mechanism design.

Detection of insider attackers.

Detection of insider attackers requires different strategies than detection of outsider attackers [108, 146]. Typically, detection focuses on the behavior of or the information created by vehicles rather than on the vehicles' identity and certification alone. For example, consider an insider attack detection strategy based on an honest majority assumption, which is commonly employed in related work [e. g., 82, 137]. A group of 100 vehicles drive on a highway in the same geographic area. Now 95 vehicles broadcast a message containing the information that traffic flows freely. Five vehicles, however, broadcast a message pertaining to a traffic jam in the same region. Receiving all 100 messages, a reasonable assumption would be that the majority of 95 vehicles report the correct situation, and the remaining 5 vehicles are attackers or have faulty sensors. This example mechanism operates on two assumptions. First, it assumes that messages from different vehicles are distinguishable. That is, so-called Sybil attacks [57] are assumed to be prevented. Second, it applies a model of traffic flow to the received messages. Following the assumption that traffic in a confined region exhibits a certain homogeneity, the 5 traffic jam reports are filtered. While

this is just one example for insider attack detection, many approaches follow a similar pattern [75, 148].

Challenges of resilient in-network aggregation.

In-network aggregation hinders these detection mechanisms, which makes resilient protocol design a particular challenge. The reason is that information is continuously modified and summarized as it is forwarded through the network. Moreover, summarization aims to reduce the amount of communicated information to prevent over-saturating the wireless medium [152]. By doing so, cryptographic signatures of atomic observations are invalidated and atomic reports about the same event become indistinguishable. In the aggregation setting, a vehicle that receives the 100 reports may create a summary and forward only the summary to farther away vehicles. How can these far away vehicles verify that the summary is consistent with the real world situation? In the exemplary insider attack detection mechanism, distinguishable reports from honest vehicles serve as "witnesses" for the correct traffic situation. But replacing such atomic reports with a single much more compact summary representation is the goal of an aggregation mechanism. By doing so, aggregation also removes the witnesses' integrity protection function. This conflict uncovers a central trade-off between in-network aggregation resilience and bandwidth efficiency.

To find solutions for this trade-off, proposals for resilient in-network aggregation in VANETs have explored a number of approaches to facilitate insider attack detection. Some proposals add additional atomic reports and cryptographic signatures to summaries, and others introduce additional redundancy [2]. But these approaches are often limited to specific traffic situations, restrict the flexibility of aggregation, or induce considerable bandwidth overhead.

It is the goal of this thesis to investigate flexible and bandwidth-efficient resilient in-network aggregation mechanisms that are applicable to a wide range of traffic situations, guided by the central research question:

Main research question.

> How can the resilience of an in-network information aggregation process and integrity of aggregated information be ensured while maintaining the bandwidth benefits introduced by aggregation in VANETs?

1.2 OVERVIEW AND CONTRIBUTIONS

In the following, we[1] introduce sub-questions, which reflect the research strategy of this thesis, and we discuss our main contributions addressing those subquestions and our central research question. The thesis structure, which follows the subquestions and contributions, is referenced throughout the discussion.

1.2.1 *Modeling vehicular in-network aggregation*

Research questions and contributions in Chapters 1 to 3.

In the design of resilient mechanisms, the first step is to understand what kinds of security mechanisms are applicable to aggregation, what limitations aggregation poses, and what aspects of aggregation mechanisms can be beneficial for security. In Chapter 3, we present a comprehensive discussion of in-network aggregation as a way to address bandwidth problems. We focus our analysis on two modeling sub-questions.

1. What are the mandatory components and functionalities necessary to build an efficient aggregation scheme?
2. How does information spread and evolve within the network?

Towards understanding the intricacies of aggregation protocols, which change and merge information during forwarding unlike other protocols, we identify typical use cases, as well as representative examples for in-network aggregation protocols. Our central contribution is

a. a *generic architecture and information flow model* for vehicular in-network aggregation mechanisms.

The architecture model addresses Question 1, and the information flow model addresses Question 2. The architecture model allows to understand the function of individual architecture components. Knowing which components exist and how they operate on information is a necessity to design suitable security mechanisms.

Modeling results have been published in [2, 3, 6, 7].

1.2.2 *Security analysis of vehicular in-network aggregation*

Research questions and contributions in Chapters 4 to 6.

In-network aggregation poses a number of unique security challenges, which we discuss based on two sub-questions in Chapter 4.

3. What are an aggregation scheme's assets and how could an attacker modify disseminated information?
4. What are approaches to ensure the resilience of aggregation mechanisms against such attacks?

Our security analysis results in a definition for resilient aggregation, including a quantitative metric to measure mechanism resilience (Section 4.3.2). We developed security goals for in-network aggregation and introduce attacker models to answer Question 3.

Based on the security goals and attacker model, we contribute

b. a *taxonomy of security mechanisms* for vehicular in-network aggregation.

1 In the spirit of humble research, and following customary writing style, the first person plural will be used throughout the thesis, despite it being a monograph.

We present cryptography, interactivity, data consistency, and trusted hardware as approaches for achieving resilient aggregation (Section 4.6), addressing Question 4. We discuss advantages and disadvantages of each approach. We consider solutions in each category, but we argue that resilient aggregation protocols should not rely solely on trusted hardware due to complex and costly management of such systems. The discussion of security approaches lays the foundation for our main mechanism proposals, which employ cryptography, interactivity, and data consistency checks.

As additional contributions towards analyzing secure aggregation, we introduce and discuss a number of challenges faced by secure aggregation protocols, and we highlight privacy implications of in-network aggregation. The security analysis is followed by a review of related work on secure in-network aggregation in Chapter 5. A key observation is that existing work on secure aggregation often focuses on fixed, inflexible aggregation functions; is limited to certain traffic situations or attacker scenarios; or produces a large amount of security overhead. Therefore, we identify the need for more flexible, efficient mechanisms that can be applied to a wide range of contexts.

Based on our taxonomy for security approaches, we therefore propose and investigate novel security mechanisms. Our central proposals address each of the promising approaches identified in answering Question 4; we discuss the methodology followed for mechanism design in detail in Chapter 6.

Results of the security analysis have been published in [2, 188, 10, 11].

1.2.3 *Resilient in-network aggregation mechanisms*

Research questions and contributions in Chapters 7 to 10.

Our four security mechanisms serve to instantiate the abstract approaches that we identified in answering Question 4. Further, each mechanism discussion entails a detailed security analysis and overhead discussion, addressing the following sub-questions.

5. What are the benefits and limitations of different approaches to achieve resilient aggregation?

6. What is the relationship between resilient aggregation scheme overhead and the achieved protection?

We propose *two cryptography-based security mechanisms* (Chapters 7 and 8), which adapt to different traffic situations, ranging from free-flowing traffic to traffic jams. Both mechanisms' overhead can be adapted to achieve different trade-offs between efficiency and security. We analyze these trade-offs and discuss ways to further improve cryptographic protection. In particular, we contribute a detailed discussion of how cryptographic tools, such as identity-based cryptography, can be applied to reduce overhead (Sections 7.6 and 8.6). Resulting from this discussion, we gain an understanding of the advantages and limits of cryptographic protection for in-network aggregation.

To apply *interactive security mechanisms* to in-network aggregation, we design a novel velocity-based clustering mechanism, which forms stable cluster

structures by integrating aggregation decisions with clustering decisions by leveraging our modeling results (Section 1.2.1). In addition to being an efficient in-network aggregation mechanism, the clustering approach serves as basis for our interactive security mechanism. We present a security mechanism, which benefits from the cluster structure and provides bandwidth-efficient resilience (Chapter 9). As a result of the mechanisms' evaluation, we identify dense traffic situations as main application area for interactive security mechanisms.

We use our information flow model (Section 3.6.4) to derive requirements for a *redundancy-based security mechanism.* Due to the repeated information merging, using redundancy is especially challenging in aggregation settings. Resulting from our analysis, we present a suitable security mechanism in Chapter 10. Evaluating the mechanism, we gain an understanding of possible filtering mechanisms and necessary overhead to apply data consistency checks to in-network aggregation.

Summarizing, the main contributions in Chapters 7 to 10 are

c. the design of *four independent resilient aggregation mechanisms,* which instantiate all promising security approaches identified in our taxonomy; and

d. their *detailed analysis and evaluation* of performance and security aspects, including a discussion of benefits, limitations, and adaptation parameters in different contexts.

1.2.4 *Mechanism combination and adaptivity*

Research questions and contributions in Chapter 11.

Due to the intrinsic trade-offs between overhead and achievable security, no single mechanism is likely to provide a perfect solution for ensuring resilience of in-network aggregation. Rather, the choice of algorithms rely on situational bandwidth constraints on the one hand and application security requirements on the other hand, leading to our final sub-question.

7. What are the implications on resilience and security overhead of combining multiple resilient aggregation schemes with individual limitations?

We present a framework to combine and adapt mechanisms in Chapter 11. The main contribution is

e. a *comprehensive resilience framework* that introduces a generalized representation of security mechanism outputs, allowing to combine and adapt mechanisms based on context.

The framework allows for mechanism selection based on traffic situations, as well as mechanism adaptation based on attack likelihood. The framework is an integrated resilience mechanism comprised of solutions from Chapters 7–10, thereby integrating their partial results to answer Questions 6 and 7. Based on the evaluation results of the individual mechanisms, we identify traffic density

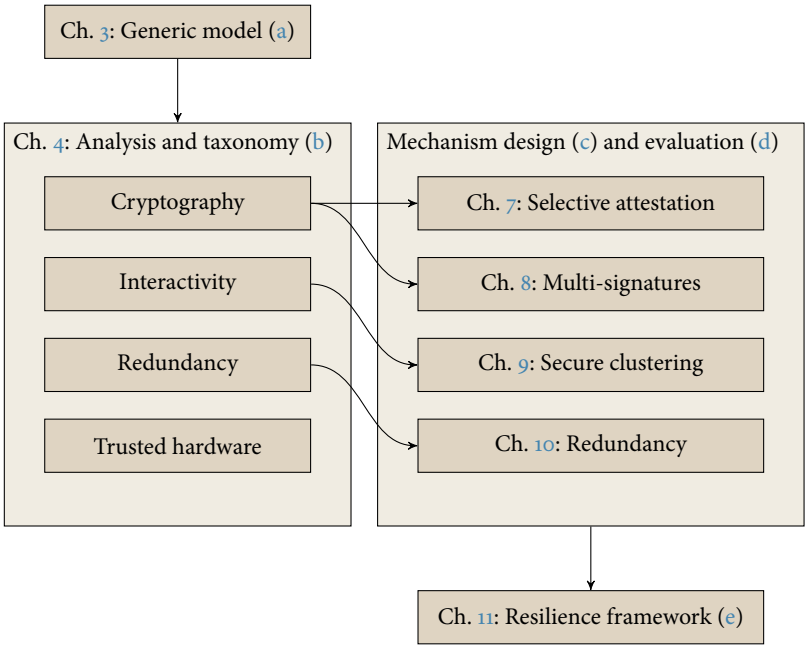


Figure 1.1: Overview of thesis chapters and main contributions (a–e).

as a suitable metric to enable or disable mechanisms. This combination allows to achieve resilient aggregation in a wide range of different traffic contexts. In addition, we use individual mechanism adaptation parameters to scale security overhead based on attack likelihood. We design a generic representation of the individual mechanisms' outputs using subjective logic, which enables combination of multiple mechanism outputs, as well as gradual transitions between mechanisms when traffic conditions change.

Results relating to security mechanisms have been published in [1, 4, 5, 9, 13, 14].

Summary of main contributions

Figure 1.1 summarizes our main contributions and methodology. We introduce a generic architecture and information flow model to enable a thorough security analysis of in-network aggregation mechanisms. As a result, we identify a taxonomy of four security approaches. We argue that cryptography, interactivity, and redundancy are the most promising approaches and introduce independent security mechanisms for each category. A resilience framework allows to combine and adapt each independent approach to implement resilient aggregation in a wide range of contexts.

VEHICULAR NETWORKS

2.1 OVERVIEW

2

Vehicular networks are a broad field of research with many possible applications, challenges, and research directions. An understanding of this underlying network and application setting is required to build resilient and scalable aggregation mechanisms. In this chapter, we therefore introduce the vehicular networks domain.

We first discuss application domains (Section 2.2) and network characteristics (Section 2.3). In contrast to the unicast-dominated Internet, message dissemination in VANETs is largely broadcast-oriented; we provide an overview of message dissemination forms in Section 2.4.

Based on understanding main VANET applications and characteristics, we introduce bandwidth issues as one of the main research challenges for applications that need to scale to large numbers of participating vehicles in Section 2.5. These bandwidth issues give rise to the research field of in-network aggregation, which we introduce in Chapter 3.

2.2 APPLICATION DOMAINS

Building on message exchange between vehicles, a wide variety of applications can be realized. Applications can be categorized into safety, efficiency, and entertainment applications.

One of the immediate VANET goals is to make driving safer. As an example, consider an emergency break warning, which is shown in Figure 2.1. Emergency break warning was identified as one of eight high potential applications by the vehicle safety communications (VSC) consortium [79, 238]. Once vehi-

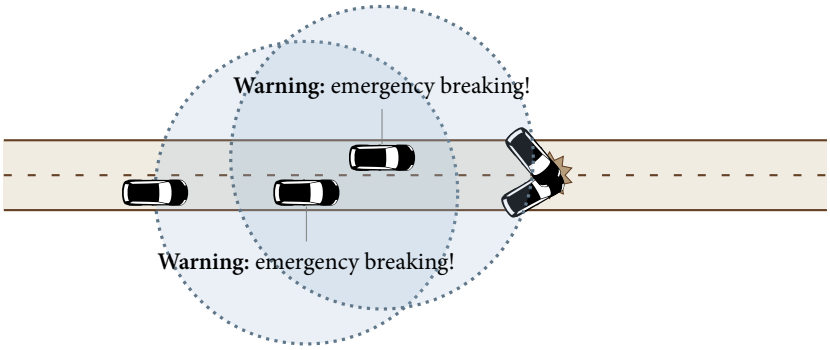


Figure 2.1: An accident on a highway. Approaching vehicles can send emergency break warnings to avoid more accidents.

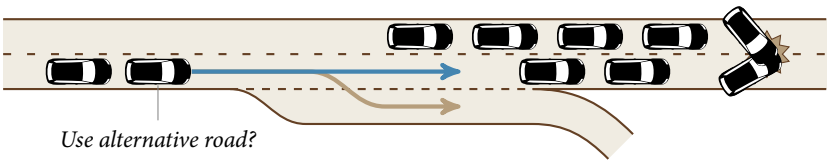


Figure 2.2: A traffic jam on a highway. Traffic efficiency applications can help approaching vehicles to decide whether to stay on the highway or take alternate routes.

cles break harder than a certain threshold, for instance due to an upcoming accident, they broadcast messages to warn approaching vehicles about their breaking maneuver. In contrast to visual warnings, such as breaking lights, these messages can be received even when visibility on the road is low due to fog. In contrast to Radar- or Lidar-based obstacle detection, messages work even if there is no direct line of sight to the breaking vehicles. The warning will be conveyed to the approaching driver using a suitable output modality, such as a warning message displayed on the dashboard or head-unit, vibrating steering wheel or audio warning. In addition to displaying warnings, the vehicle may perform automatic actions, such as breaking without driver interaction. Such automated reactions, however, pose additional legal questions concerning liability in case of system faults.

Additional safety applications.

Besides the exemplary emergency break warning, a wide range of safety applications is envisioned, including lane merging assistants, airbag warnings, and icy road and other road condition warnings. ETSI, a European standardization body that received a mandate for vehicular communication standards [234], has identified a set of foreseen basic applications [198]. Looking at communication requirements, safety applications typically require low-latency, high-frequency updates about the surrounding events.

- *Low latency* is required because vehicles might need to react quickly;
- *high frequency* is required because the event that triggered the messages might change or disappear quickly.

However, safety applications are triggered by events that are typically close to the reacting vehicle. Hence, communication over large distances, that is, several kilometers, is not required.

Improving driving efficiency.

Besides safety, one of the main VANET visions is to improve driving efficiency. Today, many cars come equipped with electronic navigation systems. Moreover, navigation systems can be purchased from a number of third party vendors [160]. Trusting these systems, drivers do not need to consult and carry paper maps and manually find routes anymore. The true potential of electronic systems, however, lies in their ability to adapt to changing conditions. Current systems use traffic message channel (TMC) [213] messages or proprietary communication using cellular networks [e. g., 206, 236] to update routes based on current traffic situations. VANET efficiency applications can use direct message exchange between vehicles to further improve routing. As an example, consider

the situation shown in Figure 2.2: a vehicle is approaching a traffic jam on a highway, but can choose to take an alternative route. Using vehicle-to-vehicle communication, the cars within the traffic jam can communicate detailed and up-to-date information about the congestion. Message exchange is more immediate than periodically fetched information from traffic centers.

Finding parking spots.

Similarly, communication can be used to collaboratively count parking spots [e. g., 42, 113, 114] or detect weather phenomena and road conditions. Also, communication between vehicles can facilitate global traffic flow optimization. Navigation systems exchange information about driver destinations, which is then used to find a global selection of routes that best uses the available road network [e. g., 26, 175]. Even infrastructure, such as traffic lights, can be considered and included in traffic optimization [e. g., 26, 67, 76]. Unlike safety applications, efficiency applications need information from large regions to calculate optimal routes. For instance, the maximum distance between two exits is 19.1 km [231] along the German highway A7, Europe's longest national highway [186]. And to consider alternative routes, information about the stretch of highway between two exits needs to be available, as well as additional information about congestion on alternative roads. Similarly, routing in cities requires knowledge about current traffic conditions. In Berlin, the total length of the road network is over 5000 km, occupied by 1.28 million registered cars; in Moscow 1.6 million cars spread over 4350 km of road [235]. Even if only a subset of the total road network is relevant for individual journeys, the geographic area and information base are considerable.

Yet, less frequent updates are tolerable, because efficiency applications change the mid- and long-term driving strategy rather than provoking short-term maneuvers.

2.3 NETWORK CHARACTERISTICS

Communication in vehicular networks is largely different from end-to-end communication, which dominates the Internet traffic [153]. The physical and medium access control (MAC) layers, derived from earlier IEEE 802.11 standards, were originally published as amendment 6 to 802.11-2007 (IEEE 802.11p), and are now part of IEEE 802.11-2012 [210]. The physical layer uses dedicated frequencies in the 5.9 GHz band [191]. Most of the time, messages sent are relevant for more than one receiver; hence, link-layer broadcast is the predominant way to send messages. Some applications require information transmission to vehicles that are not within direct communication range. For these use cases, vehicles can act as message forwarders: they re-broadcast messages that they received to their neighbors, which in turn re-broadcast them further until the intended receivers received the message.

Communication challenges and paradigms.

As all communication in VANETs is wireless, communication protocols have to cope with the absence of a reliable transmission channel. Message collisions can occur because of hidden and exposed terminals, as well as multi-path propagation effects [229, Ch. 2]. In city scenarios, shadowing effects can especially

hinder communication [119, 120]. Typical single-hop communication range for vehicular communication is expected to be approximately 250 m, ranging up to 1000 m in ideal conditions [161]. An additional challenge is high vehicle mobility. On highways in certain countries, relative speeds of up to 400 km/h are conceivable when two vehicles pass each other in opposite directions. Thus, the network topology is constantly changing, which rules out all communication protocols that depend on maintaining a known structure of vehicles, interaction between protocol participants, or explicit acknowledgements. Notable examples for such protocols are most clustering approaches, as well as tree-based message dissemination schemes, both of which are often used in wireless sensor networks (WSNs) [97]. Instead, VANETs use probabilistic, redundant communication protocols to ensure high message delivery ratios [223].

To facilitate communication, a so-called on-board units (OBUs) is installed in each vehicle. The expected processing power for OBUs is expected to be limited for cost reasons. In addition, special chips are foreseen for tasks that cannot be fulfilled efficiently enough with an all-purpose CPU. For instance, specialized hardware is likely to be used for cryptographic operations [171], such as signature verifications (cf. Section 4.2). Energy consumption of communication hardware is a lesser concern, because these devices can be powered by the rechargeable on-board battery.

These characteristics set VANETs apart from WSNs. As a typical WSN application, consider environmental monitoring [36] where a number of small, low-energy devices are deployed to measure temperature and other parameters. Processing power of a WSN node is very limited (e. g., 8 MHz), and energy conservation plays a major role in communication protocol design [155].

The unique network characteristics of VANETs influence all aspects of inter-vehicle communication, including message dissemination approaches, security mechanisms, and privacy implications.

2.4 MESSAGE DISSEMINATION PATTERNS

Throughout the thesis, primarily European standardization activities will be referred to, except where noteworthy differences exist in other countries.

Within VANETs, vehicles act as information producer and as information consumer. For instance, an emergency break warning is created by a breaking vehicle, and it is relevant for all vehicles that need to react to avoid a collision. Similarly, traffic information systems require message dissemination to a group of vehicles, as discussed in Section 2.2. These addressing forms stand in contrast to Internet routing, which is dominated by unicast transmission to specific endpoints, and to WSNs routing, which requires transmission from many sources to a single (or few) sinks [97]. Therefore, a number of VANET-specific routing and dissemination patterns have emerged.

2.4.1 *Single-hop beaconing*

As a basic communication primitive, vehicles exchange periodic messages with high frequency using link-layer broadcast. Termed *beacons*, these message con-

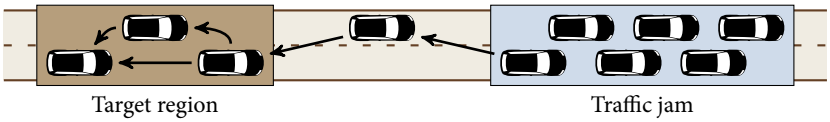


Figure 2.3: A traffic jam notification is forwarded using geocast.

tain a number of static vehicle parameters (e. g., length and width), as well as a vehicle's current position, time, and an identifier. Foreseen beaconing frequencies range from 1 to 10 Hz [199], depending on channel load. In Europe, ETSI standardized cooperative awareness messages (CAMs) to provide a beaconing service [199]; similar message standards exist in other countries [87, 88, 218, 233].

Single hop beaconing is used as a baseline for many applications, as well as for other communication patterns [153]. Receiving continuous beacons, vehicles become aware of other vehicles surrounding them. Beacons are typically stored in a world model; the corresponding ETSI standard refers to this world model as local dynamic map (LDM) [196]. One of the world model's main use cases is to support safety applications: analyzing positions of neighboring vehicles, dangerous constellations can be detected and warned about. Moreover, the world model serves to derive information for more advanced communication patterns. For instance, the number of neighbors can be used to deduct current channel load and adapt forwarding decisions, the neighbors's positions can be used to determine the best forwarding route, and so forth.

2.4.2 *Multi-hop dissemination*

While link-layer broadcast can support a number of applications, many more applications require information dissemination in larger regions. For these dissemination patterns, vehicles act not only as information observers, but also as routers, which rebroadcast information. As an example, consider traffic jam warnings on a highway. To give approaching vehicles time to react and recalculate routes, traffic jams should be known when they are still kilometers away. While we cannot identify specific vehicles that need to receive the traffic jam warning, we can define a geographic target region, which contains all vehicles that are interested in the traffic jam notification.

The geo-broadcast communication pattern.

To disseminate messages in such target regions, so-called geographic broadcasting, short geo-broadcasting or simply geocast [211, 153], is used as communication pattern. Figure 2.3 illustrates the basic concept: vehicles within the traffic jam detect the congestion, create an information message, and define a target region in which the message should be disseminated. If the vehicles that create messages are not themselves part of the target region, the message first needs to be forwarded towards the target regions. Because forwarding vehicles in this stage are not interested in the message content, geographic routing is used to

forward messages towards the target region with the least possible number of forwarding vehicles. In addition, vehicles may choose to store information and forward it once the vehicles have moved closer to the destination region. Once in the target region, the message is rebroadcasted by more vehicles to make sure each vehicle overhears the contained information. Many proposals exist to create efficient – in terms of bandwidth consumption – yet robust – in terms of message delivery rate – geocast protocols (e. g.[24]).

Besides traffic jam warnings, geocast can be used for a number of applications, including weather condition warnings and information about approaching emergency vehicles [37]. ETSI standardized a protocol to implement geocast [202], as well as so-called decentralized environmental notification messages (DENMs), which are used for event-triggered (i. e., not periodic) messages containing warnings and information and are disseminated using geocast.

Relation to infrastructure and cellular communication.

As it is explained above, geocast relies solely on communication between vehicles, making use of the ad hoc network they form. Because multi-hop communication using vehicles is especially sensitive to network characteristics such as mobility, density, and packet collisions, many researchers propose to augment pure vehicle-to-vehicle communication with road-side unit (RSU) communication or use of cellular networks, such as the UMTS or LTE.

RSUs are communication units deployed as part of road-side infrastructure and often have a connection to a backend network. Therefore, they can be used to quickly disseminate information in arbitrary geographic regions [e. g., 115, 135]. Vehicles can report their traffic information to RSUs, which are connected to centralized servers using wired network infrastructure. The server aggregates reports and disseminates current traffic information. While the deployment of RSUs is considered in densely-inhabited urban areas, estimated deployment and maintenance costs of 3,000–5,000 US dollars per RSU [25] are widely considered prohibitive for full road network coverage.

Besides the option to use RSUs, the use of cellular networks instead of multi-hop ad hoc communication has gained momentum in recent years, and UMTS coverage and LTE coverage are becoming a commodity in urban areas. When available, cellular networks can help to disseminate information to vehicles that are hundreds of kilometers away. However, cellular networks may face capacity issues similar to those of VANETs when they are used in situations with high vehicle density, such as traffic jams or intersections [118]. Moreover, cellular coverage may not always be available. Therefore, we see cellular networks as a technology that can complement multi-hop communication when it is available.

2.5 BANDWIDTH ISSUES

Because all communication in VANETs is wireless and all information is broadcasted at least to direct neighbors, messages have to compete for wireless channel capacity. On a congested, multi-lane highway, a large number of vehicles

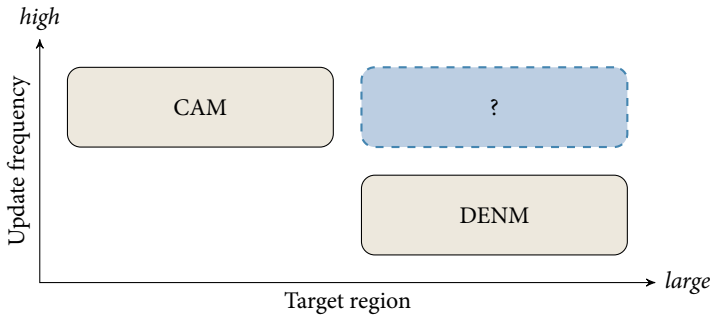


Figure 2.4: Message frequency vs. target region design space.

may be in direct mutual communication range and need to share the wireless channel. Wireless contention in these situations may be acceptable for frequent messages with small dissemination range (e. g., CAMs) and infrequent messages with large dissemination range (e. g., DENMs), as shown in Figure 2.4. However, frequent updates that need to be disseminated in large regions are problematic. Scheuermann et al. [152] have shown that for applications such as traffic jam warning systems, proper information granularity management is a necessity to cope with bandwidth limitations. We will discuss Scheuermann et al.'s results in more detail as part of the requirements analysis in Section 3.4. Here, we will use an example highway traffic jam warning application and an estimation of the induced bandwidth overhead to discuss limitations of naïve implementations and aggregation benefits.

Multi-hop bandwidth usage example.

We assume a stretch of dense traffic on a highway, and the goal is to warn upcoming vehicles about the congestion. To increase utility for navigation systems, warning messages should contain the average velocity of vehicles in the dense traffic situation. To transmit the warning to approaching vehicles, DENMs can be used. However, the DENM can only be created and disseminated once the extent of the dense traffic interval and the average velocity are known. Currently, ETSI does not standardize a way to detect such information, yet it is clear that all vehicles within the high traffic area need to collaborate to calculate its extent and average velocity.

A naïve dissemination mechanism.

First, assume a simple flooding approach is used to exchange information between vehicles. Each vehicle periodically creates a message that contains the minimum set of information necessary to collect traffic information. Messages contain the vehicle's position, a vehicle identifier, speed, and road ID. When all items are encoded as they are in ETSI's message standards [199, 201], each message consumes 25 bytes. Vehicles disseminate these messages to their direct neighbors using link-layer broadcast with a rate of 1 Hz. To achieve multi-hop dissemination, each receiving vehicle re-broadcasts received messages. However, duplicates are ignored, and messages are only forwarded once to implement a basic broadcast storm protection [130]. Eventually, all vehicles within the dense traffic area will have received information from all other vehicles.

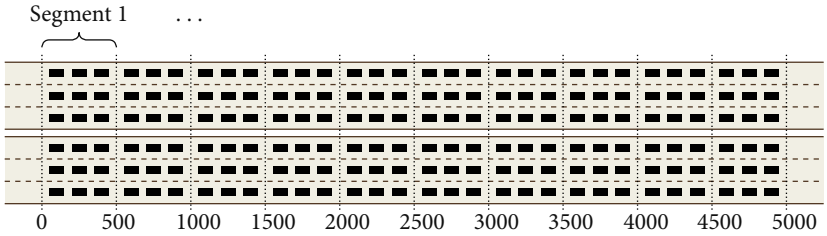


Figure 2.5: Naïve traffic jam warning application.

Then, the extent and average velocity of the congested stretch of road can be calculated and used to create DENM warnings.

As the traffic situation may change over time, the process repeats periodically, aiming to achieve a 1 Hz update rate. But as the amount of information increases with each forwarding step, the question is whether a 1 Hz update rate is feasible. To calculate the achievable rate, we divide the road into segments, as shown in Figure 2.5, which are determined by communication range. As shown, we assume a 5 km long stretch of highway with 3 lanes per direction, all congested, an average car length of 5 m, 1 m space between cars, and 250 m minimum communication radius. In each segment, 500 cars compete for wireless bandwidth. We can use the transmit time formula in IEEE 802.11 [210] to derive that *at most* 10.8 such 25 bytes messages can be (re-)transmitted per vehicle per second (cf. [61], [154]) under ideal channel conditions. However, already in segment 1, each vehicle needs to rebroadcast a total of 500 messages per second, that is, all messages received from direct neighbors, which would consume at least 46 s. Eventually, messages will need to be dropped to keep up with newly generated messages. In segment 2, the number of messages doubles, because the messages from segment 2 have to be forwarded, as well as all messages from segment 1. After 10 segments – i. e., 5 km –, 5000 messages need to be forwarded, which would consume 463 s. In order to maintain a 1 Hz update frequency, 99.8 percent of the available messages need to be dropped after 5 kilometers forwarding distance. These numbers are in line with Scheuermann et al.'s results [152], which argue that a reduction that is quadratic in the forwarding distance is required to maintain a fixed update frequency.

Intelligent message filtering is an intrinsic necessity.

When 99.8 percent of all potential information needs to be dropped and only 0.2 percent are kept, the selection criteria for the remaining 0.2 percent will influence the utility of the remaining information. For instance, when only information from the left half of the congested area is kept, the traffic jam extent will be wrongly calculated. When only those messages with the lowest velocity reports are kept, the calculated average velocity may be biased. In-network aggregation mechanisms implement distributed algorithms that filter information based on locally available context information while maintaining information utility for applications. More specifically, in-network aggregation mechanisms perform three tasks:

1. they evaluate own sensor observations and received information locally at each vehicle to detect larger events represented by clusters of similar information (e. g., congested road stretches represented by similar velocity values);

2. they combine information about those events using knowledge of information semantics (e. g., congested road stretches are represented by a single message specifying their extent and average velocity); and

3. they implement dissemination mechanisms to inform farther away vehicles about the detected events and situations.

By doing so, in-network aggregation mechanisms effectively reduce the total bandwidth usage. In contrast to other efficient dissemination mechanisms, such as efficient geocast protocols [e. g., 24], in-network aggregation requires more application knowledge to calculate suitable information summaries. But as a result, in-network aggregation has the potential to represent information in the most compact form that still offers sufficient utility for applications.

2.6 SUMMARY

VANETs are an active field of research with a wide area of possible applications. We have explained the predominant use cases and application scenarios for vehicular networks. Being a form of mobile ad hoc network (MANET), vehicular networks face unique communication challenges and exhibit specific communication patterns that are different from classical Internet routing protocols. One of the main challenges for vehicular network is the wireless medium used for communication.

In particular, its bandwidth limits the amount of information that can be transmitted within a given geographical space. At the same time the potential number of network participants is huge. While some applications only require information from vehicles in the direct vicinity, others require dissemination in larger areas. These factors combined make protocol design for vehicular networks challenging. Especially applications that source their information from a large number of vehicles and disseminate them to a large number of vehicles – such as traffic information systems – face bandwidth issues. In-network aggregation can help to address these bandwidth issues and will be introduced in detail in the following chapter.

IN-NETWORK AGGREGATION

3

Parts of this chapter (Sections 3.2 to 3.5) are based on our survey on the topic [2]; the generic model (Section 3.6) is a revised and extended version of [6, 7]; and our representative dynamic aggregation mechanism (Section 3.7.2) has been published in [3, 12].

3.1 OVERVIEW

In-network aggregation is one of the more complex information dissemination patterns in VANETs. In contrast to other patterns, information is altered during forwarding, and it is merged and summarized using knowledge of the information semantics. In this chapter, we introduce in-network aggregation as an information dissemination mechanism that copes with the bandwidth issues we discussed in Chapter 2.

The design of suitable in-network aggregation mechanisms is challenged by unique requirements of VANETs. We give an overview of requirements and challenges in Section 3.4, and we review related work on in-network aggregation in general in Section 3.5. To understand the underlying architectural concepts of in-network aggregation and the information flow during aggregation, we introduce a generic model (Section 3.6), which we exemplify using representative aggregation mechanisms (Section 3.7).

The requirements, challenges, and generic model introduced in this section will serve as the basis for our security analysis in Chapter 4, as well as our mechanisms in Chapters 7 to 11.

3.2 INTRODUCTION AND DEFINITION

As we have seen, naïve multi-hop communication quickly overloads the wireless channel, which causes messages to be dropped. Intuitively, knowledge about message semantics can help to efficiently filter messages without causing too much impact on data utility. A key observation in Section 2.5's bandwidth use example is that a large amount of messages are exchanged to make all vehicles aware of the current traffic situation. Namely, each car broadcasts its observed current velocity and position. However, only a very compact result is necessary to support navigation applications. Namely, the traffic jam's extent and average velocity. While the raw observations are necessary for the computation of the summary, only the summarized result is necessary for the application.

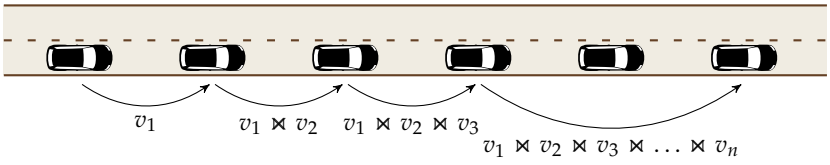


Figure 3.1: Vehicles on a highway disseminate aggregated velocity information in a larger area. Here $v_i \bowtie v_j$ denotes that v_i and v_j have been aggregated in some way, for instance, by calculating their average.

In other domains, such as WSNs, in-network aggregation has been successfully applied to reduce the number of messages [64, 129]. Applied to VANETs, the core idea of in-network aggregation is that – instead of forwarding own observations and received messages unmodified – each vehicle uses information about data semantics to summarize data items and disseminates the result to other vehicles using a single message. These other vehicles, in turn, apply the same mechanism. This process decreases the total number of messages substantially, because multiple received messages are aggregated into one. We define in-network aggregation characteristics as follows:

A definition for in-network aggregation.

DEFINITION 1 *In-network aggregation* in VANETs is any kind of multi-hop message dissemination where a number of vehicles collaborate to gain knowledge about real-world phenomena. To do so, they exchange messages containing relevant information derived from atomic sensor readings or other means of information collection. During the dissemination of information, aggregation aims to reduce the amount of redundant information by processing and modifying atomic information items.

Besides detecting traffic jams, aggregation mechanisms are suitable for a number of use cases. Aggregation especially helps applications where vehicles collaborate to analyze real world phenomena using sensory information. Such applications aim to enhance vehicles' awareness about their wider surroundings. As a result of in-network aggregation, summarization of information is performed as close to the data sources as possible, whereas, in traditional applications, data is transferred to a centralized system and only analyzed and summarized there. Applying in-network aggregation to a traffic information system, each vehicle would analyze received location and velocity information, add its own velocity observation, and calculate the velocity average and an interval that contains all locations, and only forward the result, as shown in Figure 3.1.

Benefits of semantic aggregation have been exploited in a number of applications.

Ideally, aggregation mechanisms can employ knowledge about information semantics when they merge information. In other domains, using information semantics has led to very efficient compression mechanisms, such as JPEG [215] for pictures, MPEG4 [214] for movies, and MP3 [216] audio files. In our example, knowing that messages contain velocities allows to derive that their average will suffice for navigation application decisions. Moreover, knowing that the traffic jam extent is to be determined allows to combine only messages with similar velocity and locations.

On the downside, such semantic compression is typically not invertible: once data is aggregated, the original items cannot be reconstructed without error. Therefore, careful optimization towards specific use cases is necessary to achieve the best trade-off between data utility and bandwidth usage. Next, we present applications that may benefit from in-network aggregation. From these specific applications, we derive generic characteristics that make aggregation applicable to these applications.

3.3 USE CASES

Prime use cases for in-network aggregation in VANETs are applications that collect and continuously disseminate information about the road network and surrounding real-world phenomena [2].

IN TRAFFIC INFORMATION SYSTEMS, vehicles collaborate to disseminate an approximation of the current traffic situation. Either, only the presence of an event (e. g., traffic jam) is communicated, or actual (average) speed values are communicated.

WEATHER INFORMATION SYSTEMS collect data about average temperature or severe weather conditions can be aggregated. Again, either only the presence of an event or averages can be used.

ROAD CONDITION WARNINGS apply to all forms of road conditions, e. g., stretches of icy roads or stretches with bad tarmac condition, can be summarized using in-network aggregation mechanisms.

PARKING SPACES are a mostly city-specific application. Aggregation can be used to collect the number of available parking spots in different areas.

Traffic information systems are the primary application for in-network aggregation.

Traffic information systems are by far the most cited in literature [e. g., 52, 3, 4, 66, 115, 121], followed by parking space finders [e. g., 42, 114]. Weather and road conditions are sometimes mentioned as alternative applications for proposed mechanisms, but they are never evaluated as main use cases for existing aggregation mechanisms [2]. We discuss different aggregation approaches in detail in Section 3.5.

A commonality of these use cases is that they are composed of large sets of raw sensor data, but can be represented in an abstract, much smaller, form, abstracting from the raw data but keeping all properties that are required by applications. In fact, interpretation of the raw data is *needed* both to make algorithmic decisions and to present collected data for users. For instance, a traffic information system uses as basis atomic speed observations from all vehicles. But navigation decisions are made based on abstract properties, such as estimated travel times for different roads [115]. These abstract properties are derived from the raw data. An efficient aggregation mechanism for navigation systems may summarize all raw reports but needs to maintain enough information to derive approximate travel times.

Types of detected situations.

As shown in Table 3.1, mechanisms for different use cases operate on different kinds of information: some mechanisms only detect presence of events [e. g., 82, 181], others calculate averages or count objects [e. g., 42, 3, 114]. Those mechanisms that detect events are typically only interested in the presence of certain phenomena, for instance, traffic jams or stretches of icy road. More details, like average speed within the traffic jam or exact road adhesion values are ignored. Besides the considerable information loss, another disadvantage of mechanisms that only detect events is that they often only communicate abnormal behavior. For instance, only traffic jams but not stretches of free flow

Table 3.1: Example use cases for in-network aggregation

Application	Events	Values	Mentioned
Traffic information systems	traffic jam	average speeds, minimum speeds, travel times	very often
Weather information systems	—	average temperature, average visibility	sometimes
Road condition warnings	icy road, road construction	—	sometimes
Parking spaces	—	# of free spaces	often

are communicated. As a result, vehicles cannot determine whether stretches of road about which they do not have information are currently not congested or whether they are congested but the information has not been communicated yet.

In order to improve on simple event detection, therefore, many schemes use more detailed values, such as average speed or temperature. Parking spot finders mostly use counts of free parking spots in certain regions. Moreover, some proposals for traffic information systems aggregate derived travel times instead of the underlying average speed information [e. g., 115]. Compared to average speeds, average travel times are more abstract, and, therefore, offer even more potential for bandwidth saving. On a highway where parts are congested and traffic on other parts flows freely, these facts cannot be aggregated without loss of data utility if average speeds are used. But the average time to travel a long stretch of that highway may very well still be sufficient for applications.

Because traffic information systems are the most-cited use case in literature, we will continue to use examples from traffic information systems to illustrate mechanisms we present. However, all presented mechanisms can also be applied to other use cases, such as parking spot finding, if not noted otherwise.

3.4 REQUIREMENTS AND CHARACTERISTICS

Tight integration of application knowledge and networking leads to requirements and challenges.

Summarizing the vehicular network characteristics – and especially the bandwidth limitations – discussed in Section 2.5 and the use cases for aggregation discussed in Section 3.3, the design space for efficient aggregation is bounded by bandwidth efficiency on the one hand and data utility and security on the other end. In particular, we identify four main requirements for efficient data aggregation. Security and privacy requirements, being the main focus of this thesis, will be separately discussed in Chapter 4.

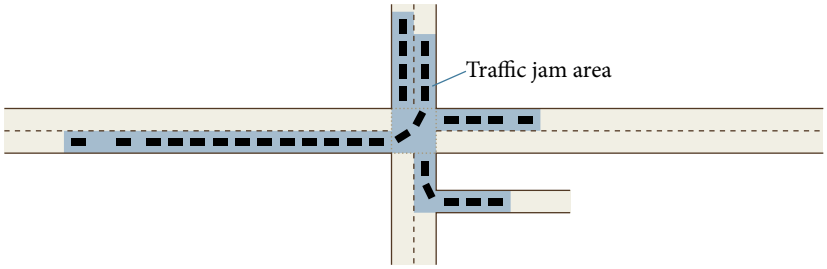


Figure 3.2: A complex traffic jam scenario in a city. Encoding the traffic jam's complex extent may lead to higher bandwidth overhead.

3.4.1 *Data reduction*

In a vehicular network, all vehicles constantly produce information using their sensory equipment. The sum of all sensed information makes up a fine-grained picture of the whole road network. Similarly, all vehicles on the road are interested in at least a subset of all collected information. Even further away information can be interesting in case a vehicle is on a long trip. Thus, data needs to be transported over multiple hops from producers to receivers. Transporting all possible information will overload the wireless channel. Therefore, data granularity needs to be reduced along the way. Simple reduction by a constant factor is not enough, because these solutions will not scale to wide range dissemination [152]. Thus, data reduction must scale with increasing distance. In particular, Scheuermann et al. [152] show that the bandwidth used for dissemination must be within $o(1/d^2)$ if d is the distance to the observed event. If more bandwidth is used, the dissemination will not scale to arbitrarily large areas. That means that at some point, the available information would oversaturate the wireless channel, which would lead to packet loss.

Here $o(\cdot)$ denotes the Little "o" notation [101].

3.4.2 *Representation overhead reduction*

As a result of data fusion and compression, aggregated messages contain information about a certain region of the road network. As information changes over time, a time interval, or any other measure for time periods, is often encoded, too. In contrast to single, unaggregated information, where location and time can be expressed by a GPS coordinate and a timestamp, descriptions of aggregated location and time can become arbitrarily complex. The exact representation depends on the aggregation scheme's application scenario If only designed for highways, schemes often use an event's start and end positions on the highway, together with a road id, to denominate the location of aggregated events [e. g., 3, 114, 170]. And they may use time intervals to state the time period in which the aggregated information has been collected.

In a city center, aggregated events can be much more complex, as shown in Figure 3.2. For instance, a traffic jam can extend across intersections to a num-

Meta-data may introduce additional overhead.

ber of roads. Semantically, the whole traffic jam may be regarded as a single event, and all vehicles' reports within the traffic jam may be aggregated. However, the description of the geographic area covered by the traffic jam will then likely be complex and diminish the aggregation bandwidth savings. As an alternative, fixed grids [169] and known, pre-shared road network maps with hierarchical segmentation [42, 45, 115] can help to achieve more efficient event descriptions. However, such pre-shared information needs to be disseminated and continuously updated for all vehicles.

As such, the problem of overhead reduction relates to a well-known trade-off in dictionary-based compression mechanisms such as LZW [184] algorithm. The more meta-information is known by receivers, in case of LZW dictionaries containing encoding symbols for often-used byte sequences, the more efficiently can data be encoded.

3.4.3 *Preservation of data utility*

Whenever data is merged, quality requirements of applications need to be taken into account. All data needs to be aggregated as much as possible while keeping its most important characteristics to support application decisions. For instance, data about a traffic jam may not be merged with surrounding data of free flowing traffic, because otherwise traffic jam avoidance algorithms will make wrong decisions. Moreover, aggregation needs to handle redundant data about the same event. This redundant data should not result in biased representations of the real world. Therefore, aggregation mechanisms should employ mechanisms to filter duplicates in sensed data [e. g., 114]. Moreover, information that changes over time needs to be considered. Only information that co-occurs, i. e., originates from the same period of time, should be aggregated. Or, if aggregates are supposed to depict development of events over time, they should contain an explanatory encoding of the time period they represent to allow receivers to correctly interpret their contained information. For instance, information about past traffic jams may provide input to heuristics that estimate likelihood of future traffic jams. Therefore, reports' time periods provide important context information.

Aggregates may be subject to bias due to duplicates.

Moreover, aggregation mechanisms should be robust against basic sensor failures or inherent sensor biases. For instance, GPS accuracy commonly fluctuates in the range of multiple meters [e. g., 168]. Aggregation functions determine the extent to which aggregation mechanisms are influenced by such uncertainties. For instance, an arithmetic average is influenced more easily by outliers than a median. In addition, aggregation mechanisms can explicitly detect outliers by analyzing and filtering information before it is aggregated.

3.4.4 *Flexibility*

The requirements discussed above imply the need for flexible aggregation decisions. Simple aggregation schemes use fixed size segments as spatiotemporal

structure. While being easy to implement and resulting in low overhead, these structures neither scale well in terms of data reduction, nor do they preserve data quality well. Better schemes use flexible data structures that adapt to application requirements and data reduction requirements at the same time. Being able to express flexible location and time intervals, more flexible schemes are intuitively more susceptible to consume more encoding overhead. Therefore, schemes have to be carefully designed to achieve a good compromise between flexibility and overhead.

In the following, we discuss exemplary in-network aggregation schemes for VANETs. The discussion is a summary of a survey we published on aggregation in vehicular networks [2].

3.5 RELATED WORK

Information aggregation as a broad concept is applied in many areas of computer science. To give an overview of different domains, we begin our related work discussion with an outlook on applications beyond vehicular networks. Based on the general overview, we then discuss different existing approaches for VANETs in detail. We focus on mechanisms that emphasize communication efficiency rather than on information integrity protection. Related work on secure aggregation will be discussed in more detail in Chapter 5.

3.5.1 *Aggregation in other domains*

Aggregation in databases.

Besides VANETs, aggregation has been successfully used in a number of other domains. Many information systems require processing, analyzing, fusing, and filtering of raw data to distill a more abstract representation, which is of better utility for applications. Traditional information systems were often designed in a centralized fashion. Information is stored in a database and query languages are used to extract information. These query languages, for instance, the well-known SQL language [217], offer methods to aggregate stored data by calculating their sum, average, standard deviation, or other statistical functions. The field of data mining [71] is driven by the idea to apply much more complex functions to derive semantics and meaning from raw data. In contrast to VANETs, such classical applications benefit from centralized data storage, which enables many analysis methods.

Still, centralized databases face problems, which led to the development of mechanisms used in many VANET aggregation mechanisms today. Namely, Flajolet-Martin sketches (FM sketches) [70] provide a mechanism to probabilistically count the duplicate-free sum of database entries in a single pass. Using multiple passes to eliminate duplicates is often impossible in VANETs due to the high vehicle mobility and short interaction times. Because duplicate-insensitivity is an important requirement, many schemes [e. g., 72, 78, 14, 114] apply FM sketches.

Aggregation in sensor networks.

The necessity for efficient and distributed information aggregation gained more momentum when small-scale devices with sufficient battery and processing power could be manufactured. Such small scale devices are used to create WSNs and other mobile ad hoc networks. WSNs consist of a large number of small, battery-powered devices equipped with a number of sensors, as well as wireless communication hardware. One of the main use cases for WSNs is environmental monitoring, which requires the collection of temperatures and other sensor values [174]. These are then forwarded using an ad hoc communication network to one or few sinks, which further analyze data and potentially trigger actions [20]. Energy consumption is a major concern for WSN, and the nodes' communication units are major energy consumers. Therefore, many proposals for networking protocols explicitly consider energy preservation, for instance, by choosing nodes with short distance to create forwarding paths [155, 158].

In-network aggregation is well-researched in the WSN domain to further reduce energy by reducing the amount of information that needs to be communicated. Exemplary surveys on in-network aggregation in WSNs have been written by Fasolo et al. [64] and Nakamura et al. [129]. Rather than providing a complete survey on WSN aggregation, we therefore focus on discussing two key differences between WSNs and VANETs, which have defining influence on in-network aggregation mechanism design:

1. sensor notes typically change their position slowly or not at all; and
2. one or few sinks in WSNs consume information generated by sensor nodes.

Because node mobility is typically limited, WSN aggregation protocols often implement distinct phases for establishing forwarding paths, information collection, and path maintenance. Because only few nodes are responsible for collecting information, trees are the predominant forwarding structure for WSN aggregation. The tree is typically created by the sink flooding an initial message through the network [e. g., 54, 116]. Receiving nodes use this message to determine their shortest path towards the sink and store their parent nodes. Thereby, a tree hierarchy is created. Later, nodes use the established paths to forward information towards the sink. Each parent node collects information from its direct children and performs aggregation. As an alternative to hierarchical trees, some protocols propose to establish cluster structures [e. g., 111]. Having created clusters and selected cluster head nodes, again the established structure is used to perform aggregation. To amount for some mobility and node failures, paths may be periodically checked.

Differences in VANET aggregation.

In VANETs, nodes – that is, vehicles – move with much higher velocities. On highways, relative speeds of 200 km/h are conceivable for traffic going in different directions. Even traffic going in the same direction may be heterogeneous. For instance, dutch highways are limited to 130 km/hfor private vehicles and to 80 km/h for trucks [241]. Therefore, relative speeds up to 50 km/h will commonly occur. Assuming 300 m communication range, vehicles will

only stay in mutual communication range for 5–20 s. Therefore, trees would need to be constantly re-created to maintain forwarding paths.

Moreover, aggregation trees require that a common root node can be identified. While this scenario is common in WSNs, each vehicle that participates in a VANET is typically both information source and information consumer. For instance, each vehicle can contribute its velocity and other parameters to a traffic information system, and at the same time, each vehicle is interested in receiving such information from other vehicles. For these reasons, VANETs require different approaches to aggregation. Specifically, tree-based and other approaches that require a fixed structure have been argued to be inefficient [e. g., 48], and have been abandoned in favor of dynamic, structureless aggregation mechanisms [2].

3.5.2 *VANET aggregation schemes*

Many different aggregation schemes for VANETs have been proposed in the past decade. We organize our discussion chronologically and highlight specific aspects of aggregation that have been addressed. These focus aspects – i. e., aggregation decisions, information fusion, and information dissemination – serve as basis for deriving a generic model of aggregation mechanism components in Section 3.6.

Fixed segments.

One of the earliest mechanisms is the *self-organizing traffic information system* (SOTIS) [169, 170], which was originally introduced in 2003. SOTIS divides the road into communication-range-based segments. Within each segment, vehicles aggregate their atomic observations and only disseminate the aggregates further. Receiving vehicles only keep the newest aggregate per road segment, but they do not aggregate the information about several segments further. Moreover, SOTIS does not implement any duplicate detection for atomic observations, which might lead to biased aggregation results. The communication overhead is reduced by a constant factor, but still grows linearly with the area that information is communicated about. Therefore, SOTIS does not scale to large areas as shown by Scheuermann et al. [152]. Following SOTIS' publication, several improved protocols were proposed.

Aggregation decisions

Hierarchical aggregation.

A number of newer aggregation schemes have improved on SOTIS' flat aggregation structure. Caliskan et al. [42] use hierarchical structures to disseminate information about free parking spots in urban scenarios. Quad trees [68] are used to represent hierarchy, as shown in Figure 3.3. The idea is to disseminate more accurate information about free parking spots in the local vicinity and only disseminate coarser, aggregated information in larger areas. Note that this aggregation process only lessens the geo-spatial granularity of information, whereas the total count of available parking spots stays accurate. For traffic information systems, this approach cannot be simply transferred, because the aggregation hierarchy describes larger and larger two-dimensional squares, whereas traf-

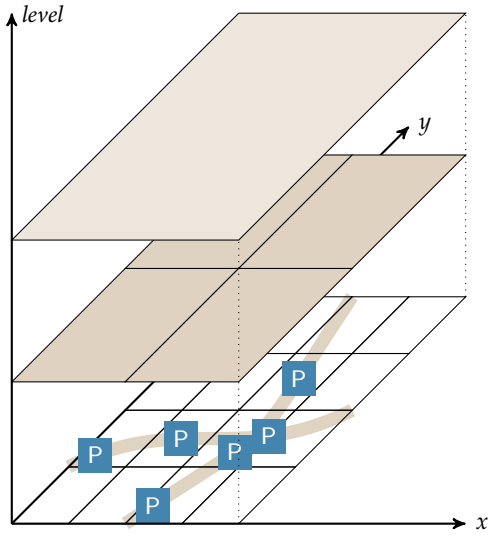


Figure 3.3: Hierarchical aggregation based on distance of observations.

fic jams and other traffic information will extend following the road network. Lochert et al. [115] define an aggregation hierarchy that is closer to the needs of a traffic information system. Instead of dividing a city according to a quadtree structure, a hierarchical set of landmarks and interconnections between them is defined. The exchanged information describes the travel times currently required between these landmarks. Travel time serves as a combined metric of speed and distance, supporting routing algorithms well. Aggregation hierarchy is represented by smaller and smaller subsets of landmarks, which describe the most recognized points of cities. Both the quad tree structure and the landmark structure are fixed in the presented schemes.

Dynamic structures.

Envisioning more dynamic aggregation structures, the TrafficView system introduced by Nadeem et al. [126] (also in [125], [51], and [127]) introduces data quality considerations for aggregation decisions. The goal of TrafficView is to disseminate traffic status information over larger areas. Aggregated information items describe clusters of close-together vehicles using a list of vehicle IDs, their averaged position, speed, and the time of the aggregate's oldest input information. To decide which items are to be aggregated, the road ahead is divided into a number of regions; region sizes are not fixed but adapt to context. Inside the regions, aggregation decisions are made using a cost-based metric. The "cost" of aggregation is designed such that it is high whenever aggregation would introduce a large error, for instance, by combining two atomic values that have a large relative distance. The idea of cost-based aggregation is used by many following papers, because it allows the aggregation system to adapt to different traffic situations. However, the specific implementation used by TrafficView still leaves the list of vehicle IDs in an aggregate as a linearly growing component. Therefore, TrafficView is only suitable for aggregating a small

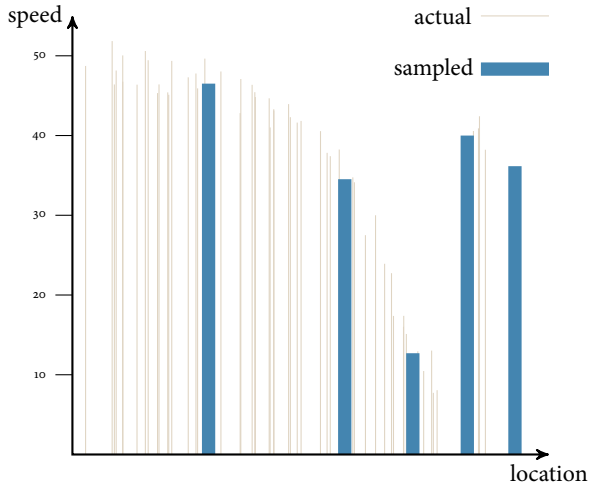


Figure 3.4: Traffic approximation using a subset of representative values. Fine lines represent all available information; only the bold values are kept and represent an approximation of the traffic situation.

number of vehicle records, increasing the visibility in the local scope. One of our aggregation mechanisms further improves aggregation flexibility by using fuzzy logic rules [3]. We describe our own approach in Section 3.7.2.

Kumar and Dave [103, 104] propose to apply ideas from multi-criteria decision making systems (MCDMs), originally proposed in the field of operations research [183], and use them to achieve flexible aggregation decisions. A k-d-tree [27] data structure is used to index vectors, which represent the similarity scores of information items. Then, application preferences and requirements are used to calculate the best aggregation strategy amongst several available strategies.

An approach that also aims at a dynamic, situation-dependent depiction of a road is proposed by Eenennaam and Heijenk [60] and improved by Schwartz et al. [156]. Borrowing ideas from run length encoding and pulse code modulation (cf. Salomon [228] and Waggener [240]), the authors try to represent the traffic situation using only a small subset of representative atomic values. To select the representatives, a threshold function is used so that only vehicles with a speed that deviates significantly from the last entry in the list add their current values. The resulting sampling approximation is shown in Figure 3.4. In order not to over-emphasize outliers, each new sample added to the list does not represent a single vehicle alone, but represents the averaged speed of a set of close-by vehicles.

Information fusion

Whereas the previous schemes mainly focus on how to structure aggregation decisions based on a road or a road network, the following schemes focus on

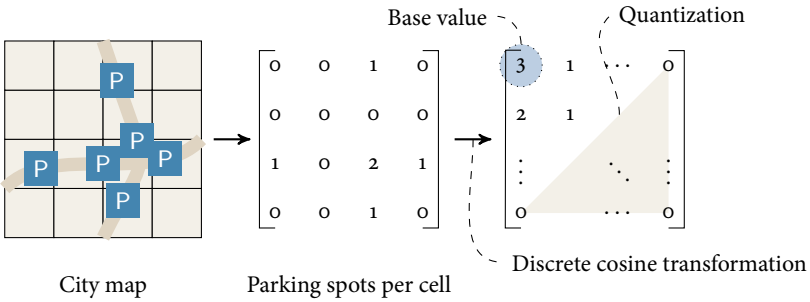


Figure 3.5: Application of a discrete cosine transform (DCT) to traffic data. From left to right: map with streets, corresponding parking spots per cell, and parking spot information after DCT transformation. After the DCT, a quantization is applied that filters high-frequency information.

mechanisms that perform the actual fusion of several information items. Fusion methods can be categorized into *syntactic* and *semantic* approaches. *Syntactic* aggregation uses techniques to compress the data from multiple vehicles in order to fit the data into a single communication packet. This compression results in lower overhead than sending each message individually. In *semantic* aggregation, the data from individual vehicles is summarized. For instance, instead of reporting the exact location of five vehicles, it is only reported that five vehicles exist [141]. While syntactic compression allows for lossless reproduction of the original data, bandwidth savings are limited by the requirement of exact reproduction. Semantic compression offers higher compression rates at the cost of information loss.

Differential encoding techniques.

Basic syntactic data compression techniques, such as ZIP [243] or LZW [166], are very generic and are therefore rarely used in VANETs. Still, approaches exist that try to achieve sufficient compression using syntactic, invertible compression. For instance, *cluster-based accurate syntactic compression of aggregated data in VANETs* (CASCADE) [83, 85] uses a variation of differential coding. Differential coding is frequently used to encode audio signals. The version used here can be seen as a syntactic version of Eenennaam and Heijenk's encoding [60]. Differential coding works by first calculating a base value, in this case median speed, and then representing all other values by their difference to the base value.

Also trying to filter out less relevant information, Zooming [45] is a technique based on discrete cosine transform (DCT) [19]. A DCT transforms a sequence of data points to the frequency domain. There, they are expressed in terms of a sum of cosine functions oscillating at different frequencies. Figure 3.5 shows the process for VANETs as proposed by Chang et al. [45]. Sensed information, like parking spots, is represented as sums per map cell. Then, DCT is used to transfer the counts to the frequency domain, which extracts the base value and differences to it, similar to differential encoding. Using quantization,

DCT is, for instance, used in lossy compression of audio (e. g., MPEG Audio Layer III (MP3) [216]) and images (e. g., JPEG).

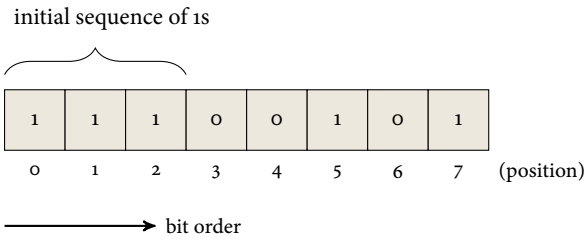


Figure 3.6: Example FM sketch. Seven items are inserted into an 8-bit long sketch, resulting in 5 bits being set to 1. The initial sequence of 1 bits, here $l = 3$ bits long, allows to approximate the number of inserted items.

information can be reduced while retaining the base value, which represents the average number of parking spots in the whole region.

Besides efficient compression, a notable problem of data fusion methods is that of duplicate messages due to redundant sensor readings. Due to the decentralized nature of aggregation mechanisms, vehicles often receive aggregates and cannot tell whether they have already contributed to the contained information. As a result, vehicles might add their own observations multiple times. The result is a biased aggregate that reduces information quality, as discussed in Section 3.4.3.

Using estimators to achieve duplicate-insensitivity.

Lochert et al. [113, 114] propose to use a modified version of FM sketches to solve the duplicate counting problem. Originally, FM sketches were used to count the number distinct elements in large databases without the need to pass through the data multiple times. The price for these features is the loss of exact counting: FM sketches provide a probabilistic approximation of the number of distinct elements. A bit field $S = s_1, ..., s_w$ of length $w >= 1$ is used as an approximation of a positive integer (see Figure 3.6). The bit field is initialized to zero at all positions. To add an element x to the sketch, the element is hashed by a hash function h with geometrically distributed positive integer output i, where $P(h(x) = i) = 2^{-i}$. The entry $s_{h(x)}$ is then set to one. The key to duplicate-insensitivity of the FM sketch is that, regardless of the number of times an identical object is inserted, the same bit in the sketch is always set. To obtain the estimated value of the sketch, the length of the first uninterrupted sequence of 1 bits l is counted. The estimate is $E = 2^l / \varphi$, where $\varphi \approx 0.77351$ is a constant. Multiple sketches can be used to further reduce estimation error. Lochert et al. [113] extended FM sketches to automatically purge old information from sketches. Zekri et al. [179] also rely on FM sketches for duplicate-insensitive data fusion. However, they do not use the adapted version, but instead keep a number of FM sketches for each road segment, which induces additional overhead.

Information dissemination

After being aggregated the data is disseminated. Central challenges for dissemination are how to select the set of forwarding vehicles and the set of target vehicles such that all interested vehicles get the information they require while only a minimal set of vehicles forwards messages to avoid too much redundancy.

Cluster-based aggregation.

Cluster-based aggregation, for instance employed by Raya et al. [145] and Saleet et al. [149], requires the election of a cluster head that will be responsible for aggregating, controlling, and sending information to members of the cluster and beyond. The cluster head appears as a single point of failure and the efficiency of the dissemination relies on the stability of the clusters. If cluster membership changes frequently, the bandwidth required for maintenance may limit the bandwidth savings of aggregation.

Structure-free approaches.

Due to the maintenance overhead of hierarchy-based solutions, other proposals use more decentralized approaches for information dissemination. A common approach is to use relevance criteria to decide whether to disseminate information in a certain region or not. A common criterion for relevance decision is the distance between target vehicles and the area that the disseminated information is about [173]. Cuckov and Song [49] propose a structureless information dissemination scheme. It works in three phases: observation, one-hop dissemination, and aggregate dissemination. Local views of vehicles are disseminated in a fixed region using geocast. At points where new information is available to be added, information is aggregated. With increasing distance to the event location, the information resolution is reduced. Similarly, Dietzel et al. [3] employ relevance criteria, similar to those proposed by Eichler et al. [62], during message dissemination. The idea is that all vehicles keep a local world model with all known information, and a set of weighing functions is introduced to prioritize the contained information. Criteria used are distance to the event, timeliness of information, and others. Whenever information is to be disseminated, only the subset of the world model with the highest weights is selected.

Dissemination based likelihood.

But even in such a relevance-based scheme, bandwidth is potentially wasted to communicate information of low utility. For instance, it is arguable whether information about regions of free-flowing traffic needs to be disseminated at all or whether it is enough to disseminate information about abnormal traffic situations, such as traffic jams. Chen [46] proposes a dissemination scheme where only vehicles with abnormal information communicate, and other vehicles remain silent. For dissemination, epidemic routing is used [239]. Of course abnormality-based schemes have an inherent drawback: a vehicle which does not receive information on a specific region cannot know whether the situation is normal, or whether communication has not (yet) succeeded. Shafiee and Leung [157] also propose an anomaly-based protocol, which only disseminates speed information when it deviates by more than a pre-defined threshold from "normal" speeds, which are pre-configured per road.

Carry-and-forward approaches.

The approaches discussed so far mostly address the issue of how to select the most suitable information for dissemination in order to provide the most

useful information to receiving vehicles. An orthogonal problem during data dissemination is how to maximize the chance that aggregatable information is received in the same time period. Reception in the same time period helps to aggregate information, because items received in the past may be garbage collected if no aggregatable information is received in time. Most schemes employ periodic beaconing using fixed beacon intervals for dissemination, which may not be optimal for the above reasons. In contrast, Catch-Up [176, 177] proposes to adaptively change the forwarding delay according to the current context. The idea is that vehicles wait before forwarding information in order to aggregate it with redundant or similar information about the same events. Data aggregation is based on fixed size road segments. Because of the introduction of a delay, Catch-up is not suitable for delay-sensitive applications.

In summary, many proposals address specific issues of aggregation in isolation. While no single protocol exists that addresses all requirements discussed in Section 3.4, there are certain building blocks that solve specific problems. For instance, FM sketches are a viable tool for duplicate-free fusion. Likewise, relevance-based dissemination allows spreading information to interested vehicles, fuzzy logic allows for intuitive representation of aggregation decision rules, and so forth. To gain a better understanding of these building blocks, we will now take a step back and analyze aggregation on an abstract level.

3.6 GENERIC MODEL

Overview of our models.

Existing aggregation mechanisms use mechanism-specific representations of aggregates and use custom data fusion and dissemination approaches. However, in order to be able to design suitable security mechanisms, we need to understand the underlying components and identify commonalities. Therefore, we propose a generic model, which fosters a better understanding of how aggregation mechanisms work internally. First, we introduce atomic observations and aggregates as information items that contain statements about parts of the road network. An architecture that captures the abstract algorithms contained in all aggregation mechanisms, as well as a representation for the information flow through the ad hoc network complement our generic model.

3.6.1 *Location and Time*

Almost all information in VANETs is associated with time (i. e., *when* the information was observed) and location (i. e., *where* it was observed); both need to be represented in aggregation schemes.

The representation of time is either a timestamp or an interval, the latter indicating that information was gathered during a period of time.

DEFINITION 2 For our model, we define *timestamps* as $T \in \mathbb{R}$, represented as seconds. *Time periods* are represented as an interval defined by two timestamps $\mathcal{T} := [s, e]$ where $s, e \in \mathbb{R}$ and $s \leq e$.

Representation approaches for locations.

For location, we can distinguish two different representation approaches: a 3-dimensional Euclidian space and a graph-based representation. The former simply represents location as a 2- or 3-dimensional GPS coordinate and is agnostic of the road network. In case information was gathered in a larger region, the agnostic approach represents the region as a rectangle or cube. The latter uses a graph representation of the road network to indicate location. Individual points can be represented as an edge of the graph (i. e., a road segment) plus a relative point indicating the position on that edge. Regions can be represented as intervals on an edge, a list of intervals on several edges, or simply by the edge ID if the region coincides with a whole road.

Specifically, we define locations as follows.

DEFINITION 3 A *location* $L := (e, a)$ is specified using its road ID and a position $a \in \mathbb{R}$ relative to the beginning of the road.

In case locations represent regions, we represent them as follows.

DEFINITION 4 A *region* is defined as

$$\mathcal{L} := (e, [a, b]), \tag{3.1}$$

where $[a, b]$ is an interval on road e.

For instance, an aggregate informing about a 5 km stretch of road on highway A7, beginning at road kilometer 200 would be represented as

$$\mathcal{L} := \{(\text{A7}, [200, 205])\}. \tag{3.2}$$

3.6.2 *Information Items*

In in-network aggregation mechanisms, statements are made about locations or regions, which are associated with timestamps or time periods. We call the more general form *aggregates*, denominating statements about regions collected in time periods. Statements about locations at a timestamp will be called *atomic observations* and are a special case of aggregates.

Structure of aggregates.

DEFINITION 5 An *aggregate* is a tuple that contains information about a specific region at a specific time period. More specifically,

$$A := (\underbrace{\mathcal{L}, \mathcal{T},}_{(1)} \underbrace{v,}_{(2)} \underbrace{q_1, \ldots, q_n}_{(3)}) \in \mathcal{A}, \tag{3.3}$$

where $\mathcal{A}$ is the set of all possible aggregates.

We distinguish three types of information in the tuple.

1. $\mathcal{L}$ and $\mathcal{T}$ are the *locator* of the aggregate. They identify a region and a time period, respectively.

2. v is the *primary* value of the aggregate. It conveys values of in-vehicle sensors or observations provided by vehicles. Examples are speed, temperature, and parking spots. In aggregates, primary values are typically the result of an aggregation function, for instance, average speed or temperature, or sum of parking spots.

3. $q_1, \ldots, q_n$ are the *auxiliary* values of the aggregate. Such values are used by many aggregation schemes to denote the certainty or quality of information after it has been aggregated. Auxiliary values can either relate the primary value (e. g., standard deviation of an average), or they relate to the aggregate as a whole (e. g., count of observations summarized in the aggregate).

Observations are defined as special aggregates.

Usually, aggregation schemes use information items from single vehicles to bootstrap the aggregation process. Such atomic observations can be regarded as a special form of an aggregate that informs about a single geographic location and time rather than a location and time interval. For some discussions, it is useful to distinguish these initial atomic observations from aggregates. Therefore, we introduce a subset $\mathcal{O} \subset \mathcal{A}$.

DEFINITION 6 An *atomic observation* is a tuple that is composed of a vehicle's local sensor value or other direct information source at one point in time:

$$o := (L, T, v, \mathbb{1}_1, \ldots, \mathbb{1}_n) \in \mathcal{O} \subset \mathcal{A}, \tag{3.4}$$

where $\mathcal{O}$ is the set of all possible observations.

Like for aggregates, L and T are a geographical and a temporal locator, respectively. But for observations, they identify a specific location and timestamp where o was observed. In addition, the observation contains a primary value v, which represents an exact sensor reading or observed value.

The auxiliary values of observations are trivial, that is, the standard deviation is 0, count of contained observations is 1, and so forth. Here, these trivial standard values are indicated as $\mathbb{1}_1, \ldots, \mathbb{1}_n$. In practical applications, the auxiliary values of atomic observations are often omitted.

Example aggregation process.

To get a better understanding of aggregates and atomic observations, consider a traffic information system. The system's purpose is to inform vehicles about average speed on different parts of the road network. To achieve that, vehicles broadcast atomic observations with speed reports, which are later aggregated and further disseminated in summarized form. To improve readability, we assume that all reports are located on a road identified as "R1" and that timestamps are given in seconds relative to a fixed starting value. Moreover, we assume that three distinct vehicles each create an atomic observation and broadcast it:

$$o_1 = (\text{(R1, 500 m)}, 10, 50\,\text{km/h}), \tag{3.5}$$

$$o_2 = (\text{(R1, 510 m)}, 20, 52\,\text{km/h}), \tag{3.6}$$

$$o_3 = (\text{(R1, 510 m)}, 30, 60\,\text{km/h}), \tag{3.7}$$

For o_1, $L = (\text{R1}, 500\,\text{m})$, $T = 10\,\text{s}$, and $v = 50\,\text{km/h}$. Here, v represents the vehicle's velocity. As stated above, we omit the auxiliary values for atomic observations. Suppose now, the traffic information calculates the merged information using all three observations. This will result in an aggregate

$$A = ((\text{R1}, [500\,\text{m}, 510\,\text{m}]), [10, 30], 54\,\text{km/h}, 4.32\,\text{km/h}, 3). \quad (3.8)$$

After merging, both the location and the time are stated as an interval: $\mathcal{L} = (\text{R1}, [500\,\text{m}, 510\,\text{m}])$ and $\mathcal{T} = [10\,\text{s}, 30\,\text{s}]$. The velocity is given as the average of the three atomic observations. Furthermore, two auxiliary values inform about the aggregate's quality: $q_1 \approx 4.32\,\text{km/h}$ is the standard deviation of the average velocity and $q_2 = 3$ indicates that 3 atomic observations have been merged to create the aggregate. Note that this specific combination of parameters – intervals, average, standard deviation, and count – is only used for this example. While many schemes employ similar mechanisms to aggregate atomic observations, more complex operations are possible and will be discussed in Section 3.6.3. Moreover, representing the number of participants as simple integer may lead to biases, as the origin of observations may not be easily determined during the aggregation process. As an alternative, FM sketches may be used, as discussed in Section 3.5.2.

Possible extensions.

Both atomic observations and aggregates can be extended to contain more than one value. For instance, consider an application that aggregates both average velocity and average outside temperature. Since most existing aggregation schemes focus on one type of value exclusively, we omitted the possibility for multiple primary values in our definitions.

Further, we need to model the knowledge base of vehicles. Each vehicle has access to some subset of all observations and aggregates. The subset is composed of the vehicle's own atomic observations and aggregates, as well as all atomic observations and aggregates received from other vehicles.

DEFINITION 7 The *world model* of a vehicle is the entirety of all information available to a vehicle x at time T, represented by a set of aggregates:

$$\mathcal{W}(x, T) := \{A_1, \ldots, A_n\} \subseteq \mathcal{A}. \quad (3.9)$$

The world model can contain several information items with overlapping geographical and temporal regions. Moreover, items can contain inconsistent information due to faulty sensors or malicious attacks. Or, inconsistencies can arise from heterogeneous velocities on different lanes of a highway segment. The world model is therefore commonly filtered before using it for application decisions or further dissemination.

3.6.3 *Architecture*

Common architecture component arrangement.

An aggregation protocol is a distributed algorithm, executed on all participating nodes, to exchange information. Characteristic for aggregation is the modification of information along the way. Even though many different aggregation

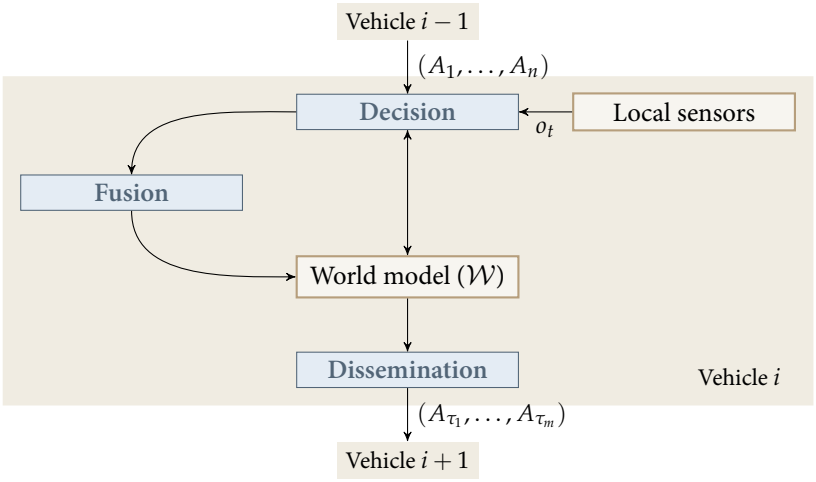


Figure 3.7: Generic aggregation protocol components *Decision, Fusion,* and *Dissemination*, which interact with the world model, local sensors, and other vehicles. Arrows represent the predominant flow of information.

schemes have been proposed, we can identify a common set of components that each scheme is comprised of. Namely, each aggregation scheme needs to *decide* whether any number of items are "similar enough" to be aggregated. Then, a *fusion* algorithm is necessary, which combines items into a joint new information item. Finally, information needs to be *disseminated* to other vehicles. Those receiving vehicles perform the same steps. Thereby, multi-hop dissemination of aggregates is achieved. In addition, a world model data structure is needed that manages all information available to a vehicle and provides efficient querying and updating mechanisms.

Existing schemes use different arrangements of these components. Figure 3.7 shows the most common architecture of aggregation mechanisms [e. g. 45, 51, 3, 126]. New information, either observations or aggregates, is forwarded to the decision component and compared with already known information. The decision component decides whether the items can be aggregated and, if so, forwards them to the fusion component. If not, they are directly added to the world model. The goal is to make sure that only items that are "similar enough" by a suitable metric are fused and other items are kept separately in order to preserve data quality. The fusion component then performs the actual aggregation and adds the result to the world model. Finally, the dissemination component selects a subset of the world model for further dissemination, creates messages, and disseminates them to other vehicles.

Alternative component arrangements.

Note that in this arrangement, new information is immediately aggregated before it is added to the world model. Because many fusion methods impose some quality loss, some authors [e. g., 113] argue that all new information should be added to the world model first. Then, the dissemination component triggers the decision and fusion process. No matter which variant is chosen, the set

of required components remains the same. We will now discuss the required functionality for each component.

Decision component

The decision component compares a number of information items (denoted $2^{\mathcal{A}}$, the power set of all aggregates) and groups similar items for aggregation:

$$\text{Decision} : (2^{\mathcal{A}}, \mathfrak{C}) \longrightarrow \{\text{yes}, \text{no}\}. \tag{3.10}$$

Besides the actual information items, the decision function is possibly influenced by context $\mathfrak{C}$, such as the current time, location, and driving direction of the own vehicle. Common decision criteria are geographical relation of information items and similarity of contained values, such as speed [e. g., 51, 3, 60]. Moreover, the decision needs to take into account temporal correlation, as well as movement direction to address the dynamic nature of VANETs. A simple decision method is to group all items from a particular road segment [169, 170]. More elaborate decision mechanisms reduce information granularity gradually with increasing distance between the deciding vehicle and the information region. Also, more complex rule sets can be used, e. g., fuzzy-logic-based decision rules [3].

Fuzzy rules are an example for complex decision techniques.

Fusion component

The fusion component performs the actual merging of information items once they have been grouped by the decision function:

$$\text{Fusion} : 2^{\mathcal{A}} \longrightarrow \mathcal{A} \tag{3.11}$$

Fusion can be a lossless or a lossy process. An example for lossless fusion is the simple concatenation of values or a difference encoding [e. g. 83]. Most existing schemes use lossy fusion in order to save more bandwidth. Several fusion functions can be used for different values contained in an information item. For instance, geographical coordinates might be fused by calculating their bounding box, that is, the smallest axis-aligned rectangle that contains all given coordinates. Speed values can be fused by calculating their average. Both examples are lossy in the sense that the original atomic values cannot be reconstructed given the result of the fusion.

Dissemination component

The decision component selects a possibly modified subset of the world model for dissemination:

$$\text{Dissemination} : (\mathcal{W}, \mathfrak{C}) \longrightarrow (A_{\tau_1}, \ldots, A_{\tau_m}) \subset \mathcal{W}. \tag{3.12}$$

In order to preserve bandwidth, only the most relevant items in the world model are selected for dissemination instead of disseminating the whole world

model. Like the decision function, the dissemination selection strategy takes into account the current context $\mathfrak{C}$ of the disseminating vehicle [cf. 62]. For instance, geographically closer information can be given preference for dissemination [3]. Also, information about traffic jams may be prioritized over information about free-flowing traffic, information about free parking spots over information about occupied parking spots, and so forth. Dissemination is most often periodic, but additional strategies, like carry-and-forward, are possible [176, 177].

Relevance-based decisions are often used for dissemination.

3.6.4 *Information Flow*

Together, decision and dissemination shape the way in which information flows through the network. Understanding the nature of this flow, especially its redundancy, is crucial for understanding how information can be altered by attackers and how information can be protected by security mechanisms. Therefore, we propose a novel model for information flow, which represents characteristic features of aggregation protocols.

To make the following discussion more readable, we first introduce an operator symbol to denominate aggregation of two or more aggregates.

DEFINITION 8 The *aggregation operator* ($\bowtie$) combines decision and will be used in infix notation. Intuitively, the aggregation operator states that the decision component selected two aggregates for merging and the fusion component has calculated a joint aggregate. More precisely, if

$$A_R = A_1 \bowtie \ldots \bowtie A_n \tag{3.13}$$

then

$$\mathsf{Decision}(\{A_1, \ldots, A_n\}, \mathfrak{C}) = \text{yes} \tag{3.14}$$

for some context $\mathfrak{C}$ and

$$A_R = \mathsf{Fusion}(\{A_1, \ldots, A_n\}). \tag{3.15}$$

Aggregation is context-dependent.

Especially note that "for some context $\mathfrak{C}$" implies that the aggregation operator is coupled with a specific point in time at which a specific vehicle has performed the aggregation. Given another context, another vehicle might have not performed the aggregation operation on the same items. As an example, consider two aggregates A_1 and A_2, which contain velocities $v_1 = 40\,\mathrm{km/h}$ and $v_2 = 30\,\mathrm{km/h}$ as primary values. A vehicle located close-by may choose to keep both aggregates separate because of the velocity difference. At the same time, a vehicle far away may choose to calculate $A_1 \bowtie A_2$, because the aggregation mechanism applies coarser decision rules to information based on geographic distance. Therefore, the aggregation operator is only properly defined if we use it within a known context.

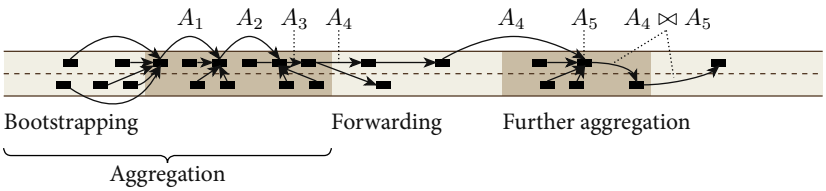


Figure 3.8: Phases in aggregation lifecycle.

Moreover, we use $A_i \in P^{\bowtie}(A_R)$ to denote that A_i is one of the information items which have been aggregated in A_R, that is,

$$A_i \in P^{\bowtie}(A_R) \Leftrightarrow \exists A_1, \ldots, A_{i-1}, A_{i+1}, \ldots, A_n : A_1 \bowtie \ldots \bowtie A_n = A_R. \tag{3.16}$$

Used by itself, $P^{\bowtie}(A_R)$ refers to the set of all information items used to calculate the aggregate A, including information items that were part of the hierarchical aggregation process.

Note that $P^{\bowtie}$ is not a preimage function in the mathematical sense, because many fusion functions remove information in an irrecoverable way. For instance, given a sum of values and the count of values summed, we cannot reconstruct the sequence of individual values. Instead the inverse aggregation function is used to illustrate information flow in the following examples.

Aggregation operator example.

To better understand the aggregation operator, consider the following example. Assume we have 3 atomic observations from three vehicles, as stated in Equations (3.5), (3.6), and (3.7). If these o_1, o_2, and o_3 are now aggregated by a fourth vehicle, which creates A as a result, we can state this as $A = o_1 \bowtie o_1 \bowtie o_2 \bowtie o_3$. The statement implies that the Decision function in the fourth vehicle has decided to aggregate the atomic observations and has applied the Fusion function. We can also say that any other vehicle calculating $A' = o_1 \bowtie o_1 \bowtie o_2 \bowtie o_3$ would have led to the same result, i. e., $A' = A$. Also, we can say that $o_1 \in P^{\bowtie}(A)$, and so forth. We *cannot* infer, however, that whenever we are given o_1, o_2, and o_3 they *will* be aggregated to A. Given other contexts, Decision functions could decide to keep them as separate observations.

Aggregation Lifecycle

In aggregation protocols, information is disseminated within the network. Because information is modified and summarized as it is forwarded, it is less trivial than in other dissemination protocols, such as geocast, to identify information items as they progress through the network. Therefore, we instead identify a certain real world phenomenon, which is the subject of the disseminated information. And we discuss the lifecycle of all information items used in the

collaborative detection of the phenomenon. We use a traffic jam as an example, but similar lifecycles can be identified for other phenomena, such as free parking spots or weather. A traffic jam can be described by an aggregate

$$A = ((e, [a, b]), [s, e], 0\,\mathrm{km/h}, c) \tag{3.17}$$

where e is the road ID, $[a, b]$ and $[s, e]$ are the traffic jam's location and time, and c the number of vehicles in the traffic jam. The aggregation lifecycle consists of 4 phases, as shown in Figure 3.8.

BOOTSTRAPPING First, individual vehicles broadcast their atomic observations o_i to direct neighbors. As vehicles receive more and more atomic observations from their neighbors and generate observations using their local sensors, they start aggregating them (A_1). Because A_1 is again broadcasted, the vehicles become aware that all neighboring vehicles are part of the same traffic jam, but the complete extent is still not known.

Depending on the implementation of the Dissemination function, the bootstrapping process can either be disorganized – as shown in Figure 3.8 – or organized. In the former case, all vehicles start aggregating as soon as they receive more than one atomic observation; in the latter case, all vehicles broadcast their atomic observations, but only specific vehicles perform the aggregation.

AGGREGATION Vehicles farther away receive A_1, combine it with atomic observations in their vicinity, and calculate A_2 and subsequently A_3, which contain information about a larger part of the traffic jam. As the process continues, at some point the aggregate will become stable (A_4). In this context, "stable" means that A_4 represents the whole traffic jam and adding more atomic observations would add negligible additional information.

FORWARDING Once aggregates are stable, they are disseminated further to potentially interested vehicles. Depending on the implementation of the Decision function, the aggregate is either disseminated in all directions to make the vehicles within the traffic jam aware of its extent, or the aggregate is only forwarded upstream to inform approaching vehicles. During the forwarding phase, aggregates are not modified anymore.

FURTHER AGGREGATION As the aggregate is forwarded further, the context of receiving vehicles changes. At some point, to save bandwidth, Decision functions may perform coarser aggregation. Hence, A_4 is fused with a close by traffic jam, represented by A_5, which was forwarded separately at first.

Inter-relation of lifecycle phases.

Because aggregation is a highly dynamic process, all 4 phases can overlap. In particular, each vehicle will periodically generate new atomic observations with newer timestamps, which may re-initiate the aggregation process for a new

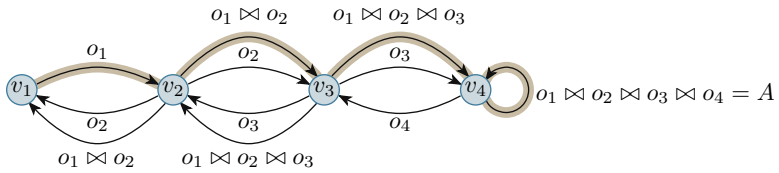


Figure 3.9: Example information flow graph with main dissemination direction highlighted.

time interval. While an aggregate about the traffic jam representing the time interval $[s, e]$ is already in the forwarding phase, an aggregate representing the same traffic jam but during $[s, e + k]$ can still be in the aggregation phase. Moreover, aggregates can switch back and forth between the forwarding and further aggregation phases. Therefore, many aggregation schemes do not make explicit distinctions between the different phases. Instead, aggregates enter each phase implicitly. For instance, the Decision function can be implemented such that aggregates are simply not fused with local atomic observations if their primary values are too different or auxiliary values, such as standard deviations, become too large. Still, making phases explicit can help to assure data integrity, as will be discussed in Chapter 4 and used in a specific mechanism in Chapter 8.

Aggregation Operations and Information Flow

As indicated in Figure 3.8, the information flow of all information items used for creating an aggregate A can be regarded as a graph. We now formally introduce this graph and explain how its properties relate to characteristics of aggregation protocols. Given an aggregate A, an observer that overheard all communication in the network can recreate A's creation process using the set $P^{\bowtie}(A)$. Regular vehicles participating in the aggregation cannot typically reproduce the process, because they only have partial knowledge of information items, and the Fusion function typically cannot be inverted.

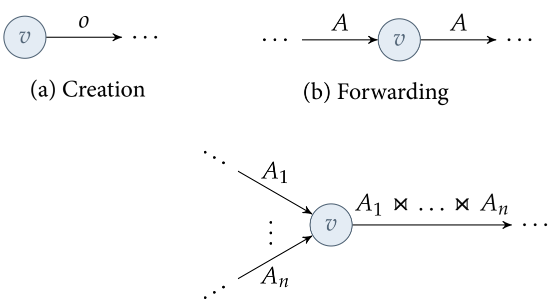
DEFINITION 9 The *information flow* leading to an aggregate A is a directed multi-graph $\mathcal{I}_A = (\mathcal{V}, \mathcal{F}, \omega)$ where

- $\mathcal{V}$ is the set of all vehicles;
- $\mathcal{F} = \{\{(v_i, v_j) : v_i, v_j \in \mathcal{V}\}\}$ is the multiset of directed edges representing messages that are transferred from v_i to v_j; and
- $\omega : \mathcal{F} \longrightarrow P^{\bowtie}(A)$ is a function that annotates edges with transmitted information items.

We use $\{\{\cdot\}\}$ as notation to distinguish multisets from regular sets.

Example aggregation information flow.

For example, assume we have a network consisting of 4 vehicles $v_1, \ldots, v_4$ where v_1, v_2 are in mutual communication range, as are v_2, v_3 and v_3, v_4, as shown in Figure 3.9. All vehicles broadcast an atomic observation o_i to all direct



(a) Creation (b) Forwarding (c) Discarding

(d) Fusion

Figure 3.10: Possible information flow subgraphs.

communication neighbors. Moreover, v_2 broadcasts $o_1 \bowtie o_2$ and v_3 broadcasts $o_1 \bowtie o_2 \bowtie o_3$. Finally, v_4 internally adds its own observation o_4, resulting in

$$A = o_1 \bowtie o_2 \bowtie o_3 \bowtie o_4. \tag{3.18}$$

Adding parenthesis to make the aggregation process more explicit, we have

$$A = ((o_1 \bowtie o_2) \bowtie o_3) \bowtie o_4. \tag{3.19}$$

In general, we can identify distinctive patterns, which characterize aggregation operations within the information flow graph. Figure 3.10 shows these patterns. We can use these aggregation operations to further better understand the graph representation and to deduct rules for the annotation function ω.

CREATION represents that a new atomic observation is created by vehicle v and communicated to other vehicles (see Figure 3.10a). The observation o is created by v if, and only if, there is an outgoing edge in $\mathcal{F}$ that ω maps to o, but there is *no* incoming edge from another vehicle with such a mapping.

FORWARDING means that a vehicle received an information item A and forwards it unmodified (see Figure 3.10b). That is, the Decision function has decided to keep the received A separate from other information items, yet the Dissemination function has deemed A important enough to be disseminated further. In the information flow graph, forwarding is characterized by an incoming edge in $\mathcal{F}$ for which a mapping to A exists in ω, as well as an outgoing edge with the same mapping.

DISCARDING happens when a vehicle's Dissemination function does not further disseminate information items, because they are irrelevant to neighboring vehicles according to the current context (see Figure 3.10d). The information flow graph represents discarding by an incoming edge in $\mathcal{F}$ for which a mapping to A exists in ω, but no such mapping exists for an outgoing edge.

(a) Tree-based aggregation (b) Dynamic, structure-less aggregation

Figure 3.11: Different aggregation flow graph structures.

FUSION is the key differentiating feature of a specific aggregation protocol's information flow graph (see Figure 3.10c). Here, a vehicle v takes several information items, which may be received from other nodes or observed by v itself, and applies the aggregation operator, i. e., performs Fusion after a positive Decision. In the graph, fusion is represented by 2 or more incoming edges, mapped by ω to $A_1, \ldots, A_n$, for which a single outgoing edge exists that maps to $A_1 \bowtie \ldots \bowtie A_n$.

More complex information flows can be built by combining the above basic operations. We propose to use the graph to derive properties of aggregation mechanisms. For example, an aggregation mechanism that uses *no* hierarchical aggregation will result in information flow graphs where the length of the longest edge path is 1. Likewise, a graph with cycles points to an aggregation mechanism that suffers from duplicates biasing the aggregation result.

Using information flow to understand redundancy.

Most importantly, however, the information flow helps to understand the dynamic and redundant nature of aggregation [10, 11]. In a network where node positions are stable, such as WSNs, the information flow can be organized as a tree: nodes can be grouped into clusters according to their geographic position and aggregate information within their groups. Then group leaders forward the results to more central nodes, and so forth. Eventually, the overall aggregation result is calculated at the top of the tree, as shown in Figure 3.11a.

In VANETs, however, positions are not stable. As shown in Figure 3.11b, many schemes discussed in Section 3.5 disseminate information in a way that makes the information flow emerge organically rather than structured. Because information is often transmitted periodically and to cope with possible collisions during transmission, the information flow will contain redundant information. The amount of redundancy in the information flow is an important metric, because, on the one hand, redundancy wastes bandwidth due to duplicate transmission of information, but on the other hand, redundancy helps to cope with transmission failures and false information due to attackers. Implications of the information flow on data consistency checks will be further discussed in Section 4.6 and Chapter 10.

Summary

In-network aggregation is a highly dynamic process, and many different proposals have been made to approach it. Still, all protocols work on similar information items, which we can represent in a generic way. Moreover, aggregation

protocols can be decomposed into their main components Decision, Fusion, and Dissemination. Identifying these components helps to better understand the focus and shortcomings of existing protocols. Extending the vehicle-centric architecture view to a view on the whole network, we can identify phases in the aggregation lifecycle and further break them down to specific operations and information flow properties, which further help to analyze requirement fulfillment, and which are the basis for some security mechanisms.

3.7 REPRESENTATIVE AGGREGATION PROTOCOLS

Based on our discussion of related work and the generic models we identify two representative approaches to implement aggregation protocols: fixed structures, such as fixed road segments, and dynamic aggregation with flexible segmentation. While no single protocol is currently being standardized, many protocols fall in either of those categories. Therefore, we will introduce a representative protocol for both categories in the following. We explain both schemes' data structures using an exemplary single road, but both schemes can be extended to apply to multiple roads by adding road ids to their aggregate and atomic observation representations. The representative schemes will be consistently used from here on to exemplify and evaluate protocols and security approaches.

FIXED SEGMENTS The fixed segments protocol, hereafter called `FIX`, is representative for all mechanisms that mainly use location as the Decision criterion for combining information. In the simplest case [e. g., 169, 170], segmentation is defined by wireless range.

The benefits of `FIX` are easy implementation and a straight-forward communication pattern. Moreover, aggregated views for small segments converge quickly, which can be beneficial for security mechanisms. On the downside, the bandwidth saved by `FIX` is a constant factor of the total bandwidth, which is commonly-agreed to not scale well [152].

DYNAMIC AGGREGATION The dynamic aggregation protocol, which we will call `DYN`, represents dynamic, structureless aggregation protocols. Here, the aggregation Decision is influenced by a number of factors, including similarity of primary values and maxima for auxiliary values, and is optimized to maintain data utility while preserving as much bandwidth as possible.

Due to `DYN`'s completely dynamic nature, the communication pattern and information flow are less intuitive. Continuously evolving aggregates make the design of security mechanisms a challenge. However, the bandwidth savings are typically larger than those of `FIX`, because information is hierarchically aggregated.

In the following, we describe `FIX` and `DYN` in more detail.

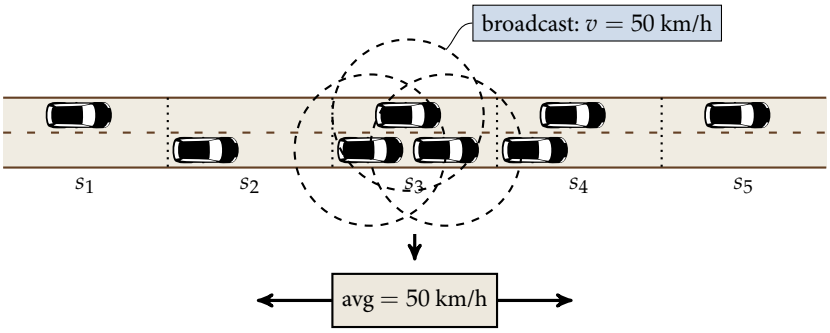


Figure 3.12: Overview of FIX. Each vehicle broadcasts the exact speed to direct neighbors; average speeds per segment are disseminated further.

3.7.1 *Fixed Segments (FIX)*

The fixed segments scheme – FIX– is based on SOTIS [169, 170] by Wischhof et al. The core idea is to impose a fixed segmentation on the road network, which correlates with the wireless communication range, and to only disseminate information with segment granularity in larger areas. Each segment is uniquely identified by a segment ID together with a road ID; both of which are assumed to be globally known. Figure 3.12 shows how FIX works. In their 1-hop neighborhood, vehicles send periodic beacons containing their current position p relative to the road's origin, road ID r, a timestamp t, and the current velocity v; the observation tuple is

$$o := ((p, t), v). \tag{3.20}$$

Receiving vehicles calculate the average velocity v of all speed reports of one road segment s to decide on a road segment's traffic status; the aggregate tuple is

$$A := ((s, t), v). \tag{3.21}$$

All such aggregates are again disseminated periodically. In contrast to individual vehicles' data, aggregates are disseminated over multiple hops. To save storage space, receiving vehicles only keep the newest summary report per road segment, assuming that this is the most accurate one. In our implementation of FIX, the main aggregation scheme components are implemented as follows:

DECISION Atomic observations are selected for aggregation if and only if their geographic identifier is in the same road segment, as indicated by the `GetSegment` function (see Algorithm 1). Aggregates are selected for further aggregation if and only if their geographic region (i. e., road segment) is the same. A mix of proper aggregates and atomic observations is not selected for fusion, because new atomic observations will be merged to a new aggregate rather than updating existing aggregates.

Algorithm 1: FIX:Decision($A_1, \ldots, A_n, \mathfrak{C}$)

INPUT: A set of aggregates $\{A_1, \ldots, A_n\} \subset \mathcal{A}$ and the current context $\mathfrak{C}$.
RESULT: yes or no, indicating whether the set of aggregates is selected for fusion.

```
IF GetSegment(A_1) = ... = GetSegment(A_n) THEN
    IF A_1, ..., A_n ∈ O ∨ A_1, ..., A_n ∈ A \ O THEN
        RETURN yes
    ELSE
        RETURN no
    END
ELSE
    RETURN no
END
```

Algorithm 2: FIX:Fusion($A_1, \ldots, A_n$)

INPUT: A set of aggregates $\{A_1, \ldots, A_n\} \subset \mathcal{A}$.
RESULT: An aggregate A that represents the merged data of all aggregates.

```
IF A_1, ..., A_n ∈ O THEN
    A ← ((GetSegment(p_1), GetCurrentTime()), 1/n Σ_{i=1}^{n} v_i)
ELSE
    A ← A_{arg max_i(t_i)}
END
RETURN A
```

FUSION Atomic observations are merged by creating a new aggregate for a road segment (see Algorithm 2). The function `GetSegment` is used to determine the fixed segment ID corresponding to a given position. The time stamp is set to the current time. All atomic speed values are averaged. Aggregates are not merged further; given two aggregates, the fusion function will drop the older aggregate.

DISSEMINATION Atomic observations are disseminated only if they were created by local sensors; the `GetSegment` and `GetCurrentTime` functions represent the extraction of sensor values from the current context $\mathfrak{C}$ (see Algorithm 3). In addition, a fixed number of aggregates (determined by a threshold `Threshold`($\mathfrak{C}$)) that represent surrounding road segments are disseminated.

As discussed in Section 3.5, fixed segments have been superseded by more flexible, hierarchical decision and fusion mechanisms. Still, `FIX` serves as an intuitive-to-understand aggregation mechanism, a baseline for other mechanisms, and it is used as underlying aggregation protocol by some security mechanisms due to its predictability.

Algorithm 3: FIX:Dissemination($\mathcal{W}, \mathfrak{C}$)

INPUT: The world model $\mathcal{W}$ and the current context $\mathfrak{C}$.
RESULT: A world model subset $A_{\tau_1}, \ldots, A_{\tau_n}$.

$X \leftarrow ((\mathsf{GetLocation}(\mathfrak{C}), \mathsf{GetRoad}(\mathfrak{C})), \mathsf{GetTime}(\mathfrak{C}), \mathsf{GetVelocity}(\mathfrak{C}))$
FOREACH $A \in \mathcal{W}$ DO
  IF $|\mathsf{GetSegment}(A) - \mathsf{GetSegment}(\mathfrak{C})| < \mathsf{Threshold}(\mathfrak{C})/2$ THEN
    $X \leftarrow X \cup A$
  END
END
RETURN X

3.7.2 *Flexible Aggregation (*DYN*)*

The flexible scheme – DYN – is based on a fuzzy-logic aggregation scheme that we present in [3, 12]. In DYN, aggregation decisions are based on a number of factors selected to optimize data utility. The main decision factors are geographic closeness of information items (independent of road segments), velocity difference, and acceptable standard deviation. Once two aggregates are selected for fusion, their combined location and time intervals, as well as the average velocity, corresponding standard deviation, and number of participants are calculated. Dissemination happens periodically, and is implemented by selecting the most relevant aggregates based on the current context for further dissemination.

Atomic observations are represented by a position $p \in \mathbb{R}^+$, timestamp $t \in \mathbb{Z}$, and velocity $v \in \mathbb{R}^+$ as follows:

$$o := ((p, t), v). \tag{3.22}$$

Aggregates use location ($[a, b]$) and time ($[s, e]$) intervals as locator, their contained average velocity is indicated by v, and they use the velocity standard deviation (ς) and number of summarized observations c as auxiliary values:

$$A := (([a, b], [s, e]), v, \varsigma, c). \tag{3.23}$$

Using fuzzy logic rules for aggregation decisions.

The key difference to the FIX scheme is the use of a fuzzy logic rule system [242] for aggregation decisions, whereas FIX bases aggregation decisions purely on the location of reports. We will briefly introduce the fuzzy logic functionality before we outline the decision, fusion, and dissemination components. In earlier work [3], we provide a more detailed discussion of fuzzy logic application to aggregation mechanisms. The main reason for choosing fuzzy rules was that a number of factors should influence aggregation decisions and their interrelations can be complex. For instance, two information items should not be merged if their velocity is too different, yet two items with the same velocity should only be aggregated if their location is close and they both originate from the same time period. While it is feasible to express these interrelations in words, their exact algorithmic representation, including numeric metrics for "too different" or "close" can be difficult.

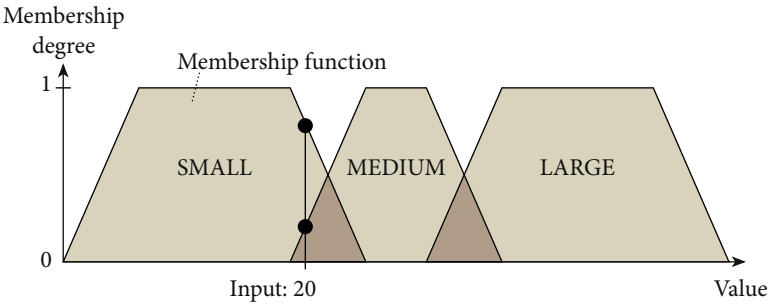


Figure 3.13: Example fuzzy logic variable *location difference* with membership functions SMALL, MEDIUM, and HIGH. An example input of 20 maps to two membership functions, SMALL and MEDIUM, with different membership degrees.

The main benefit of fuzzy logic rules is that they separate the semantic description of decision rules from their numeric implementation. First, we identify all factors that influence the aggregation decision, such as velocity difference, location difference, and time difference. All such factors can be represented as a numeric value. For use in a fuzzy rule system, these inputs are represented by linguistic variables, as shown in Figure 3.13. Each linguistic variable is characterized by a set of membership functions, which map to adjectives. As shown in the figure, an input value can map to several fuzzy membership functions at the same time but with possibly different percentages. Once input values are fuzzified, that is, they are mapped to linguistic variables, their interrelation can be expressed by fuzzy logic rules, which are if-then-statements. Consider the following example rule:

if VELOCITY_DIFF is *small* and LOCATION_DIFF is *small*
then DECISION is *yes*

In this example, VELOCITY_DIFF, LOCATION_DIFF, and DECISION are linguistic variables and *small* and *yes* are membership functions. Unlike the other variables, DECISION is an output variable, which is only assigned values as a result of fuzzy rule evaluation. As shown in the example rule, logic operators, such as "and," can be used in fuzzy rules, which are an extension of the boolean operators that take percentages as input. For instance, the fuzzy "and" of two operands $a, b \in [0,1]$ can be defined as $\min(a,b)$. As a result of the rule evaluation, DECISION is assigned a fuzzy value based on the values of the membership functions used in the condition, as shown in Figure 3.14. Once all fuzzy rules are evaluated, the output variable DECISION is defuzzified by evaluating which of its membership variable is assigned the highest membership percentage.

The minimum interpretation is the most common; other interpretations include the multiplication of a and b's percentages (i. e., $a \cdot b$) [242].

In the rule system itself, the numeric complexity of the input values' interrelations is hidden. However, the system still needs to be configured with a proper mapping between crisp input values and corresponding membership functions. For the DYN scheme, we assume a fixed mapping to be configured at

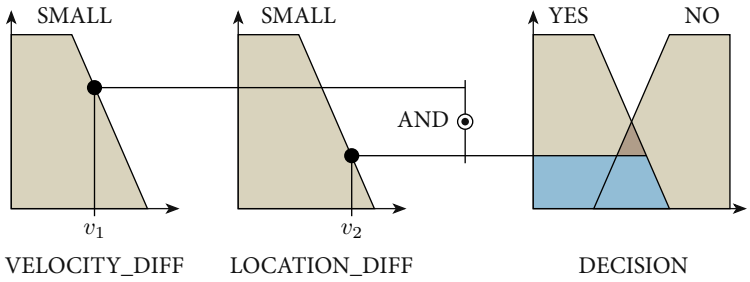


Figure 3.14: Fuzzy rule evaluation example. Two inputs v_1 and v_2 are mapped to two linguistic variables. The AND operator takes the minimum of both assignments and maps it to an area within the DECISION variable's YES membership function.

Algorithm 4: DYN:Decision($A_1, \ldots, A_n, \mathfrak{C}$)

INPUT: A set of aggregates $\{A_1, \ldots, A_n\} \subset \mathcal{A}$ and the current context $\mathfrak{C}$.
RESULT: yes or no, indicating whether the set of aggregates is selected for fusion.

location_diff ← Fuzzify(IntervalDistance($[a_1, b_1], [a_2, b_2]$)) time_diff ← Fuzzify(IntervalDistance($[s_1, e_1], [s_2, e_2]$)) velocity_diff ← Fuzzify($|v_1 - v_2|$)
IF *location_diff is small AND time_diff is small AND velocity_diff is small* THEN
  decision ← yes
ELSE
  decision ← no
END
RETURN Defuzzify(*decision*)

design time. However, the system could be improved with fuzzy rule learning systems, which optimize mapping between input values and membership functions at runtime. Employing fuzzy rules as described above, DYN's Decision, as well as the other components, are implemented as follows.

DECISION Information items are selected for aggregation if they satisfy a number of data utility criteria (see Algorithm 4). The underlying idea is to base the aggregation decision on flexible similarity rules, as explained above. For DYN, we assume a single fuzzy logic rule that still captures the idea of creating dynamic road segmentation.

FUSION Information items are merged by creating a new summary record that encompasses the location and time of all input items (see Algorithm 5). In addition, the new average velocity, standard deviation, and participant count are calculated.

DISSEMINATION Periodically, a fixed amount of information items is disseminated, which are selected using a relevance function (see Algorithm 6). Relevance is determined by an average of the items' distance to the current vehicle, the time difference, and the average velocity. The idea of the

Algorithm 5: DYN:Fusion($A_1, \ldots, A_n$)

INPUT: A set of aggregates $\{A_1, \ldots, A_n\} \subset \mathcal{A}$.
RESULT: An aggregate A that represents the merged data of all aggregates.

$A \leftarrow ((\lfloor \min_i a_i, \max_i b_i \rfloor, \lfloor \min_i s_i, \max_i e_i \rfloor), \mu(v_1, \ldots, v_n), \varsigma(v_1, \ldots, v_n), c_1 + \cdots + c_n)$

RETURN A

Algorithm 6: DYN:Dissemination($\mathcal{W}, \mathfrak{C}$)

INPUT: The world model $\mathcal{W}$ and the current context $\mathfrak{C}$.
RESULT: A world model subset $A_{\tau_1}, \ldots, A_{\tau_n}$.

$\mathcal{W}' \leftarrow$ sort A_i according to decreasing relevance using $\mathfrak{C}$.
$X \leftarrow \emptyset$
FOREACH $A_i \in \mathcal{W}'$ DO
  IF $|X| <$ MaxAggregates() THEN
    $X \leftarrow X \cup A_i$
  END
END
RETURN X

velocity is that information about traffic jams has a higher relevance than information about free-flowing traffic.

The dynamic scheme captures important characteristics of most recent proposals for aggregation schemes [2]. Namely, the (maximum) extent of aggregates cannot be pre-determined. Moreover, there is no defined endpoint of the aggregation process, after which aggregates are declared stable and not modified further. While these properties are beneficial – and partly necessary – for aggregation efficiency, they make it more difficult to devise efficient security mechanisms, as we will discuss in Chapters 7 to 10.

3.8 SUMMARY

In-network aggregation in VANETs is one of the most challenging information dissemination patterns in an already challenging network setting. The main use cases are large-scale applications that can tolerate approximate information, traffic information systems being a prime example. Unlike other dissemination patterns, such as single-hop link-layer broadcast and geocast, no single aggregation mechanism has been picked up by standardization so far. Yet, the need for aggregation has been acknowledged in the literature [152] and standardization bodies [200, Annex C.3] alike. While more aggregation mechanisms exist for WSNs and aggregation is commonly used in databases, these mechanisms cannot be applied to VANETs without modification; some mechanisms cannot be transferred at all due to the different network setting.

ETSI included a note about possible "event correlation" in its technical specification for DENMs [200, Annex C.3] but later removed it when DENMs became a European norm [201].

In this chapter, we analyzed use cases and requirements for aggregation. After presenting seminal work on VANET aggregation, we derived a generic model that dissects aggregation into its fundamental components Decision, Fusion, and Dissemination and provides insight into the information flow. Two representative protocols – `FIX` and `DYN` –, which are derived from the related work and our generic model, serve to exemplify common approaches and to compare different approaches to securing aggregation in the remainder of this thesis.

II

SECURITY ASPECTS OF AGGREGATION

SECURITY ANALYSIS

4

4.1 OVERVIEW

In in-network aggregation, information is altered and merged during forwarding. Further, every node can possibly be the proponent for information about large regions of the road network. Both properties have implications for security aspects of aggregation mechanisms. Intuitively, security protection – and especially integrity protection – is made more difficult by the algorithms' collaborative nature. On the other hand, non-invertible aggregation is beneficial for privacy, because detailed information about single nodes stays in their local vicinity and is not communicated further.

To extend the analysis of security implications beyond these intuitive notions, we will first review current standardization activities regarding baseline security mechanisms. As part of cooperative intelligent transportation systems (cITS) standardization, ETSI published a vulnerability analysis of basic message dissemination patterns [194]. In their report, however, ETSI focuses on CAM and DENM messages. Here, we rather focus on goals and threats specific to aggregation. We describe security and privacy objectives and goals and develop a formal understanding of securing in-network aggregation mechanisms. The abstract definition is then mapped to a specific attacker model and respective threats and vulnerabilities of aggregation mechanism components.

Based on the security analysis we provide a categorization and initial discussion of different approaches to achieve integrity in in-network aggregation. We conclude our security analysis by summarizing a number of challenges for security that are specific to in-network aggregation. We will present and evaluate specific security mechanisms that address these challenges in Chapters 7 to 10.

4.2 BASELINE SECURITY ASSUMPTIONS

Public key infrastructure is a requirement for protecting VANET communication.

Basic protection mechanisms for VANET communication have been the focus of a number of research projects (cf. [98, 132]) and are currently being standardized in the US [89] and Europe [192, 193]. In addition, the Car-2-Car Communication Consortium (C2C CC) – an industry-driven organization dedicated to foster wide-spread deployment of cITS in Europe – describes a possible public key infrastructure (PKI) architecture for use in vehicular networks [227]. While these standards are insufficient to protect in-network aggregation, they provide the basic primitives that serve as basis for more complex security mechanisms. In the following, we will give a brief overview of security assumptions as they are mandated in Europe's ETSI standards.

Essentially, the goal of ETSI's security architecture is to provide integrity protection against outsider attackers. To this extent, each vehicle receives cryptographic key pairs that allow to distinguish between vehicles and other non-

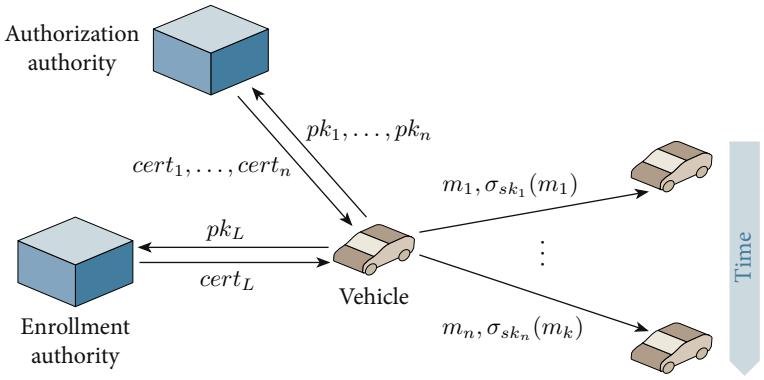


Figure 4.1: Overview of ETSI's security architecture.

authorized entities. Key pairs can be used to sign outgoing messages, providing a basic integrity protection. Once a vehicle signed a message, other entities cannot alter message content without invalidating the signature. Besides excluding outsiders without access to key material, signing all messages provides the foundation for many more complex security approaches. For instance, a number of signed messages about the same event can be used to differentiate the senders of the information item. If information is received from a number of different senders, majority votes can be used to detect attackers. We will discuss more examples and provide a categorization of security approaches in Section 4.6.

Figure 4.1 shows an overview of ETSI's proposed security architecture. When a vehicle is newly registered, it first generates a long-term key, which is an elliptic curve DSA (ECDSA) [93] 256 bit key pair (sk_L, pk_L). The vehicle then interacts with the *enrollment authority* to obtain a certificate

$$cert_L = (A, \sigma_{sk_{EA}}(pk_L, A)) \tag{4.1}$$

We use $\sigma_{sk_X}(m)$ to denote a cryptographic signature created by entity X on message m.

on its long-term key. As part of the enrollment process, the enrollment authority checks the vehicle holder's identity and other attributes (A) of the vehicle, such as its type, manufacturer, size, and so forth. These attributes, as well as the holder's identity, become part of the certificate.

For privacy reasons, vehicles do not use their long term key during communication with other vehicles [139]. Instead, the vehicle generates a number of short term keys (sk_i, pk_i) and uses its long-term key to obtain a number of short-term certificates $(cert_i = (A', \sigma_{sk_{AA}}(pk_i, A')))$ from the *authorization authority*. During this authorization process, the authorization authority checks whether the vehicle is eligible using the provided long-term key. The created short-term certificates contain similar attributes as the long-term certificate, but they do not allow receiving vehicles to link the pseudonym to a long term identity.

Outside ETSI standardization, the authorization authority is commonly referred to as pseudonym provider.

Vehicles then use the obtained pseudonyms to sign all generated messages. Due to the ad hoc nature of VANETs, vehicles also attach the pseudonym's public key and certificate to each message. It is considered infeasible that vehicles

acquire the necessary public keys and certificates for checking received messages from a centralized authority. The basic format for a message signed with pseudonym i is:

$$\left(m, \sigma_{sk_i}(m), pk_i, cert_i = \left(A', \sigma_{sk_{AA}}(pk_i, A')\right)\right). \tag{4.2}$$

Note that the certificate is commonly defined to include the public key pk. We choose to write pk and $cert$ separately, because certain omission and compression techniques treat both values differently. Of course, the certificate is bound to a specific pk due to the contained signature.

Receiving vehicles first check the validity of $cert_i$ using pk_{AA}, which is stored in each vehicle. If the certificate is valid, the signature $\sigma_{sk_i}(m)$ can be checked using the attached pk_i. In certain intervals, vehicles change their pseudonym to prevent other vehicles from tracking their movement (cf. [56]). To be able to do so, vehicles pre-load a number of pseudonyms. To prevent Sybil attacks [57], pseudonyms are equipped with a validity period. To increase driver privacy, however, validity periods typically overlap. Hence, vehicles can use few (e. g., 3) pseudonyms at the same time, but not all pseudonyms they pre-loaded. In the remainder of this thesis we assume that a pseudonym scheme may be used, but that it offers protection against Sybil attacks in the manner described above, except noted otherwise. Consequently, we will use the term "a vehicle signs a message" to mean "a vehicle uses one of its pseudonyms to sign a message."

Overhead due to signatures.

Note that attaching a signature, public key, and certificate to each message incurs a significant overhead. According to an ETSI report [193], the size of a signed header for a periodic cooperative awareness message (CAM) is 221 bytes, excluding the actual message payload. To reduce overhead, the standard proposes to attach the full sender certificate and public key only once per second and replace it with an 8 bytes digest of the certificate in the remaining messages, reducing the message size to 96 bytes. More aggressive omission techniques have been proposed, as well [41, 65, 154]. Their basic assumption is that vehicle neighborhood remains fairly stable during short time periods, so certificates can be omitted unless sudden changes are sensed. However, such optimizations only work for single-hop messages such as CAMs. Multi-hop messages, the predominant dissemination form for aggregation, reach vehicles further away. Hence, it is unlikely that sender certificates have been cached. Consequently, ETSI does not foresee certificate omission for DENMs, which are also disseminated over multiple hops.

Insider and outsider attackers.

The baseline security assumptions establish a security perimeter that allows to distinguish between *insider attackers* and *outsider attackers*, for which we use the definitions in RFC 4949 [232]:

- "An *inside attack* is one that is initiated by an entity inside the security perimeter (an *insider*), i. e., an entity that is authorized to access system resources but uses them in a way not approved by the party that granted the authorization." [232] In particular, a VANET insider attacker possesses at least one certified key pair, which he can use to create and send messages.

— "An *outside attack* is initiated from outside the security perimeter, by an unauthorized or illegitimate user of the system (an *outsider*)." [232] In VANETs, outsider attackers are entities that use arbitrary hardware to initiate attacks; in particular, they do not possess a certified key pair issued by the PKI.

The discussed PKI and signatures offer protection against outsider attackers. Protection against insider attackers using plausibility checks is considered in the respective ETSI standard [192], but no specific mechanisms are proposed. In particular, no integrity protection mechanisms suitable for aggregation mechanisms have been proposed in current standardization efforts. Proposals from researchers exist and will be discussed in detail in Chapter 5.

4.3 RISK MANAGEMENT

4.3.1 *Discussion of security objectives*

Aggregation schemes support applications that disseminate up-to-date, but not real time, data in larger areas. Thus, attacks will usually affect traffic efficiency rather than human lives. Nevertheless, reduced traffic efficiency can have a significant economic impact. It is therefore important to ensure that an aggregation scheme provides accurate information both under normal conditions and under attack. That is, the scheme should be *resilient* against attacks. In addition, the common security goals *availability*, *integrity*, and *confidentiality* need to be taken into account in addition to *privacy* requirements.

Resilience as main security goal.

Resilience means that an aggregation scheme should provide an acceptable level of service in the presence of faults, such as faulty sensors, as well as under malicious attacks. Countermeasures should be adaptive to the severity of an attack. The system should dynamically react to attacks of different severity in a way that gracefully degrades performance while maintaining normal system operation as much as possible. After attacks, the system should be able to fully recover. Resilience is the overarching goal of secure aggregation schemes.

This definition of resilience includes *availability*, the requirement that the system's service should be available at all times. An attacker should not be able to disrupt the aggregation scheme by disrupting the communication channel, dropping packets, or introducing false information into the system. Note that availability is a fundamental problem of the underlying wireless communication channel, and numerous effective attacks on availability of 802.11 wireless networks have been demonstrated [e. g., 16]. In the following sections, we will restrict ourselves to discuss only those attacks on availability that are specific to aggregation schemes.

Protecting information integrity.

It is necessary that an aggregation scheme provides accurate data to all participants and that modified data can be detected. Therefore, data *integrity* needs to be guaranteed. The central challenge for integrity protection in aggregation schemes is that data is not transferred through the network without being altered. Nodes constantly add own observations to aggregates they receive or

combine multiple aggregates and further disseminate the result. However, all allowed data modifications are clearly constrained by the aggregation scheme. Integrity protection in this context means that nodes should only be able to introduce information into the network that either stems from own sensor observations or is the result of applying the correct aggregation scheme operations to remote observations and aggregates. Any alterations to received data that do not conform to the protocol should be detected. Likewise, aggregates which claim to provide information but are maliciously crafted instead of being based on allowed calculations should be detected.

Information provided by an aggregation scheme is available to all nodes and does not need to be restricted to a certain subset. Therefore, *confidentiality* is not a goal of secure aggregation schemes. While *privacy* is a concern, most aggregation protocols intrinsically help to protect privacy of users [188]. Because aggregation protocols typically apply lossy fusion functions, vehicles typically do not disseminate exact observations in larger areas. As aggregation is a completely distributed process, no centralized entity collects all atomic observations, and consequently, no central entity can derive exact movement patterns for single vehicles. Tracking on a local scope is possible in the direct vicinity of vehicles where atomic observations are available. But it can be argued that such local tracking is a generic problem of VANET applications, because many applications rely on local dissemination of exact information.

We therefore argue that resilience, which is achieved by data integrity protection and graceful degradation of information quality in face of larger and larger numbers of insider attackers is the main security goal we pursue in protecting in-network aggregation mechanisms.

4.3.2 *A definition for resilience*

Having identified resilience as the main security goal for aggregation, we will now develop a formal definition for resilience of aggregated information. As basis, we start with a review of message integrity in an outsider attacker scenario, extend it for insider attackers, and then extend it for in-network aggregation as defined in Chapter 3. While the resilience definitions for outsider attackers and insider attackers in a non-aggregation setting are not directly applicable to in-network aggregation, they serve to reiterate important differences, as well as to make the main definition easier to understand due to its incremental construction.

Integrity protection against outsider attackers.

First, we assume a setting where two parties, a source (S) and a destination (D), communicate. The source S has a key pair (sk_S, pk_S), and D knows pk_S. S generates a message m and, using its secret key sk_S, generates a signature $\sigma(m)$ and sends $(m, \sigma(m))$ to D. An example for this setting is the dissemination of an emergency break warning message. An attacker that controls the communication channel [55] between S and D cannot modify m without invalidating σ. Hence, the message's *integrity* is protected. The semantic notion of integrity here is that D already knows S and accepts whatever S writes as correct. So D

is not concerned about the possibility that S could put wrong content in m but rather about the possibility that unknown entities could modify the message during the communication process.

DEFINITION 10 We say the integrity of m is protected against outsider attackers if for any attacker A that modifies m, the signature σ becomes invalid.

Insider attackers.

Next, we analyze integrity in a setting with multiple sources for an event but without aggregation. Assume that D wants to acquire knowledge about some real-world phenomenon that D cannot witness or sense on its own. For example, D wants to know whether there is a traffic jam down the road. For the moment, we assume that a traffic jam is just an event that either occurs or not and not a complex object that is described by the actual velocities within the traffic jam. Furthermore, we assume a number of sources $\mathcal{S} = \{S_1, \ldots, S_n\}$ that can directly observe the event. Some subset $C \subset \mathcal{S}$ of the sources can be compromised by an attacker A, and the attacker can create signatures for all sk_{S_i} where $S_i \in C$. All sources sign a message $m_i \in \{0, 1\}$ which indicates whether the event is observed to be present or not. Let $m \in \{0, 1\}$ represent the actual presence of the event in real world. All honest sources $S_i \in \mathcal{S} \setminus C$ report the correct value $m_i = m$, but the attacker-controlled sources $S_j \in C$ report an attacker-chosen value. Here, the semantic notion of integrity is different. The original anchor of confidence is the actual situation in the real world, which can be observed but not altered. But all sources that are controlled by the attacker can lie about their observation. Therefore, their signatures do not immediately make D confident that the signed message content is correct. But the attacker can create at most $|C|$ signatures, because he cannot compromise the key material of honest vehicles. Therefore, signatures protect integrity in the sense that the attacker's influence is bound by the number of signatures he can create using different secret keys. When combined with a majority voting scheme, the attacker can be detected as long as the majority of sources is honest. For instance, let

$$m := \begin{cases} 1 & \sum_{i=1}^{n} m_i > n/2, \\ 0 & \text{otherwise.} \end{cases} \tag{4.3}$$

Then m will *likely* reflect the correct real world state if the majority of sources are honest. By introducing a majority vote, the integrity is not protected purely by cryptography but instead by a combination of cryptography and a model about the real world. Here, the model is simple; it assumes that an event in the real world can be sensed by a number of sources. For the model to work, the event needs to remain unchanged for a certain amount of time so that enough honest sources can observe the event and send a report about it to the destination. In practice, models can be more complex. The consequence for the definition of integrity is that integrity is not absolute anymore. The attacker can always alter the messages of his controlled sources. Moreover, the correct reconstruction of m might fail if the majority of sources is honest, but their observation is faulty. Therefore, we use *resilience* rather than integrity to reflect

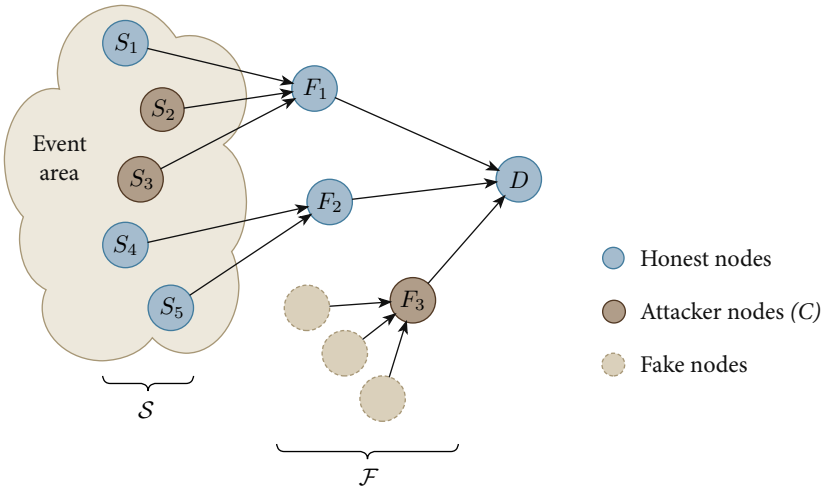


Figure 4.2: Possible points of attack in an in-network aggregation scheme.

that the mechanism cannot protect all messages against modification, but at least offers a good approximation of the real state given an honest majority.

DEFINITION 11 An information m, which is the state of an event in the real world, is *resilient* against insider attackers if there is a scheme that calculates an estimate m' using a number of sources $S_1, \ldots, S_n$ and their messages $m_1, \ldots, m_n$ such that the probability that $m' = m$ is high if the majority of sources is honest.

Resilience of in-network aggregation.

The scenario when using in-network aggregation is similar to the insider attacker scenario with two additions. First, in-network aggregation as we defined it in Chapter 3 aims to summarize arbitrary values instead of yes/no events. That is, the real world situation could be the average speed on several road segments instead of just the presence of a traffic jam. Second, information is modified by aggregation decisions and fusion on the way from the source to the destination.

Figure 4.2 shows the scenario for in-network aggregation. A destination D wants to acquire knowledge about a state in the real world, such as the traffic situation on a stretch of road. Further, D cannot observe the stretch of road itself. Instead, a number of sources $\mathcal{S} = \{S_1, \ldots, S_n\}$ observe the real world state. In addition, a number of forwarding nodes $\mathcal{F} = \{F_1, \ldots, F_n\}$ forward, discard, or fuse the sources' information, as discussed in Section 3.6.4. The attacker controls a subset of both sources and forwarders $C \subset \mathcal{S} \cup \mathcal{F}$. Instead of messages, the sources now create observations (cf. Definition 6) about the real world. Without loss of generality, we assume for the following discussion that observations have the form

$$o = (L, v) \in \mathcal{O}, \qquad L \in [0,1], v \in \mathbb{R}. \tag{4.4}$$

That is, a single value is observed and the locator is a normalized one-dimensional geographic location. We omit time in this model by assuming all observations are made at the same point in time. Using aggregation decision and fusion, forwarders create an aggregated view using the observations. For instance, F_2 in Figure 4.2 creates an aggregate $A = s_4 \bowtie s_5$. Observations may be cryptographically signed. But as soon as the Fusion function is applied, the original signatures become invalid. The reason is that information fusion creates a semantic summary of the original observations, which changes their representation. Only the aggregate itself can be freshly signed by the forwarder performing the aggregation. That fresh signature, however, cannot validate the aggregate's history. That is, it does not validate the original observations used during aggregation.

The destination receives a subset of the original observations, as well as a number of aggregates $a_1, \ldots, a_m$. Let X be the set of all observations and aggregates that D receives. The goal at the destination is to recreate the state of the real world using information from X. Bhaskar et al. [28] have formalized insider-secure aggregation for WSNs in a simpler setting where sources only report one bit and symmetric cryptography is used for integrity protection. We adapt and extend their definitions for the VANET aggregation scenario explained above.

See Section 5.2 for a detailed discussion of Bhaskar et al.'s work in the context of WSNs.

Let $R : [0,1] \to \mathbb{R}$ be a function that represents the actual situation in the real world by mapping a location $L \in [0,1]$ to a velocity $v \in \mathbb{R}$. The destination tries to interpolate this function using the received information X. Let R'_X be the function that describes D's interpretation. Even when all sources and forwarders are honest, R'_X will be an imperfect recreation of R, because of incomplete or summarized information. We describe the amount of error as the surface between the two functions:

$$\Delta(R, R'_X) := \int_0^1 |R'_X(l) - R(l)|\, dl. \tag{4.5}$$

An insider attacker can basically alter R'_X in two ways. He can change observations of sources under his control, and he can change aggregates of forwarders under his control.

DEFINITION 12 Let $\Phi(C, R'_X)$ be the function that models the altered situation created by an attacker that controls a set C of nodes, depending on the interpretation R'_X, which factors in the actual subset of information X received by the destination.

We use R'_X, which we already introduced for Equation (4.5), to represent the imperfect representation using information gathered from a set of forwarders X. In addition, the function Φ is influenced by the set of attacker-controlled vehicles C. All vehicles in C alter observations and aggregates to skew the real world representation. The more vehicles the attacker controls – i. e., the larger C, the more $\Phi(C, R'_X)$ deviates from R'_X.

Given this definition, $\Delta(R'_X, \Phi(C, R'_X))$ quantifies the attacker's influence on the real world interpolation. Note that we cannot apply Definition 11 when attackers control forwarder vehicles, because even an attacker controlling one forwarder can – given the right position in the network – arbitrarily influence the destination's interpolation R'_X. For instance, in Figure 4.2, F_3 can change D's interpolation despite the majority of forwarders and sources being honest.

As discussed in the insider attacker setting without aggregation, we also have to assume that attackers can alter the observations of sources under attacker control. But the attacker's influence using this strategy is limited. The arbitrary alteration of aggregates, however, should be prevented. Therefore, we say that the integrity of aggregated information is protected if the attacker's influence is not significantly higher than that of an equally powerful attacker that only changes observations but not aggregates.

DEFINITION 13 *An in-network aggregation scheme is resilient* against insider attackers if

$$\Delta(R'_X, \Phi(C, R'_X)) \leq \max_{C' \in \mathcal{S}, |C'|=|C|} \Delta(R'_X, \Phi(C', R'_X)) + \varepsilon \tag{4.6}$$

and

$$\max_{C' \in \mathcal{S}, |C'|=|C|} \Delta(R'_X, \Phi(C', R'_X)) \leq \Delta(R, R'_X) + \delta \tag{4.7}$$

for small constant ε and δ. Here, C' represents the maximum impact that an equally powerful attacker can achieve when compromising observations from sources only rather than being able to alter aggregates.

In summary, Definition 13 assumes an attacker that controls $|C|$ vehicles. The two inequalities make statements about the maximum influence that an attacker may be able to achieve in a resilient protocol. Equation (4.7) characterizes the maximum influence of an attacker that modifies only observations may have. We take the maximum influence that any attacker with $|C|$ vehicles chosen from the set of sources may have and argue that it should differ by at most a small constant value δ from the deviation $\Delta(R, R'_X)$ that the aggregation protocol itself introduces. Based on that, Equation (4.6) requires that an attacker that controls arbitrary vehicles, including forwarders, should at most be able to have the same influence as an equally powerful ($|C'| = |C|$) attacker that only controls sources and not forwarders.

Insider attackers may always alter observations.

The definition acknowledges that insider attackers will always be able to alter their own observations and, thereby, make the aggregation result deviate to a certain extent from the actual situation. However, the deviation by changing observations only is assumed to be small if honest nodes are in the majority, as expressed by Equation (4.7). An attacker that may change aggregates, however, can have a much higher impact and should be the main target for integrity protection, as expressed by Equation (4.6). Combining both equations, we require that an insider attacker in a resilient in-network aggregation protocol should

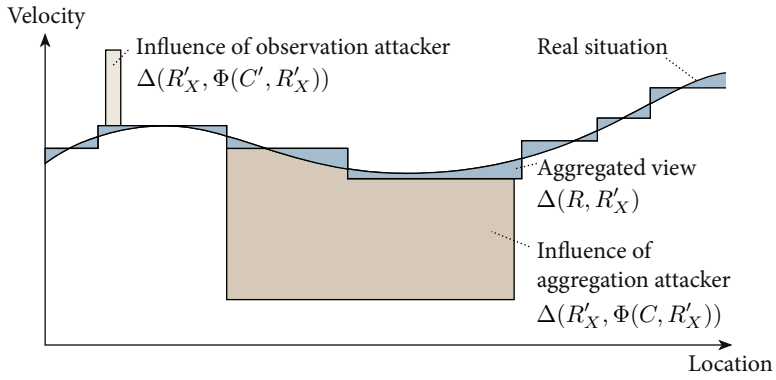


Figure 4.3: Influence of an attacker on in-network aggregation.

only be able to influence the real world representation $\Delta(R, R'_X)$ by a small constant factor $\varepsilon + \delta$.

Figure 4.3 shows the differences between the real world situation R, the aggregated interpolation R'_X without attackers, as well as the influences of two attackers that only change observations ($\Phi(C', R'_X)$) or observations and aggregates ($\Phi(C, R'_X)$), respectively. We will use this quantification of attacker influence for evaluating different security mechanisms (see Chapter 6). In the following, we will map the abstract definition of integrity protection to specific attacker capabilities and strategies.

4.4 ATTACKER MODEL

Next we describe the attacker model, which will be used as the underlying model for reasoning about threats, associated vulnerabilities. and respectively proposed security mechanisms in the remainder of this thesis. In 2015, no VANET hardware has been deployed apart from initial field operational trials. Moreover, the types of applications that in-network aggregation is most applicable to will likely not be part of the initial deployment. Due to the lack of deployed systems and reports about attacks on such systems, we discuss a wide range of possible attack vectors in the following.

4.4.1 *Capabilities*

As discussed in Section 4.3.2, an insider attacker is more threatening to in-network aggregation mechanisms than an outsider attacker. To qualify as insider, an attacker needs to be able to generate valid signatures on arbitrary messages. To generate such signatures, attackers have a number of options. Namely, the attacker can

1. trick the enrollment authority into certifying a generated long-term key;

2. trick the authorization authority into certifying a number of generated pseudonyms;
3. extract a valid long-term key from a vehicle;
4. extract a number of valid pseudonyms from a vehicle;
5. manipulate software in the vehicle to alter messages, which are then signed by the vehicle's security entity; or
6. manipulate sensors in the vehicle to alter sensor readings, which are then used to assemble messages and signed by the security entity.

Attack goals 1 and 2 target the security architecture [192], whereas attack goals 3, 4, 5, and 6 attack parts of the communications architecture [190]. Moreover, attack goals 1 and 2 require the attacker to compromise a centralized process, which we assume to be properly protected against such attacks. Likewise, attack goals 3 and 4 require significant effort, because the key storage part of the vehicles' communication architecture is assumed to be protected by a hardware security module (HSM). Messages that need to be signed are handed to the HSM, which returns a signature on the messages [203]. Key material will be stored in the HSM's protected storage and is assumed to be never accessible from the outside [190]. Therefore, attack goals 3 and 4 can be considered unlikely, except if lower-price vehicles will be deployed without HSMs for cost reasons.

However, it is generally assumed [e. g., 190, 132, 141] that only key management, storage, and cryptographic operations are covered by the HSM. Protecting the whole network stack and applications using trusted hardware would be too costly. Therefore, we make the assumption that an attacker is able to manipulate software in a vehicle in a way that allows to generate arbitrary messages and have them be signed by the vehicle's security component (attack goal 5). Likewise, not all vehicles will be equipped with tamper-proof sensors [171], so attackers could compromise sensor values (option 6). Considering attacks on in-network aggregation mechanisms, falsifying sensor readings only allows to generate false atomic observations, but manipulating other software allows an attacker to generate arbitrary aggregates.

Most likely attack goals.

Both likely attack goals (5, 6) require physical access to a vehicle. We assume that attackers cannot use remote protocols to extract key material from vehicles or have arbitrary messages be signed by remote vehicles. As discussed in Section 4.2, we further assume Sybil attack protection to be in place for the pseudonym system. Therefore, the attacker is bound by the number of cars he has complete physical access to times the number of pseudonyms he can use in parallel. We therefore assume that the majority of vehicles in any given situation is likely to be honest.

4.4.2 *Attacker goals*

The high level goal of attackers is to alter the values disseminated by an aggregation scheme. Value alteration can be arbitrary or targeted. The former means that an attacker tries to influence values to deviate from the correct view as much as possible, but does not care in which direction they are influenced. The aim here is to cause failure of applications based on aggregated information or at least confusion and lowered acceptance of the users. Targeted attacks aim to manipulate the communicated information in a specific way that is beneficial for the attacker or causes harm to other drivers. In the following, we discuss a list of specific attacker goals for the different use cases we discussed in Section 3.3.

Use case traffic situation

- *Influence navigation decisions.* An attacker tries to introduce a deviation from the correct speed information for a particular part of the road network to influence routing decisions of other drivers. For instance, the attacker tries to claim a traffic jam on a road with free flowing traffic to convince other drivers to take alternate roads. The attacker is then left with an empty road. Similarly, an attacker can claim free flowing traffic on a congested road so that drivers stay on the main road despite congestion. The results are lowered trust in the system by other drivers and less traffic on alternate roads for the attacker.

- *Increase danger of accidents.* Similar to the previous scenario, an attacker could claim a congested road stretch ahead of a target vehicle. The goal in this case is to solicit dangerous driving maneuvers, e. g., erratic breaking, due to the nearing (fake) congestion. Note that, while technically possible, this attack is unlikely. In case of a close traffic jam, other applications, which are implemented with harder real-time constraints than aggregation schemes, will warn drivers about the upcoming situation in the close vicinity.

Use case weather and road conditions

- *Alter reported conditions to influence driver behavior.* An attacker causing deviation in reported weather conditions can cause drivers to drive too fast or too slowly. The former can lead to an increased number of accidents, the latter can lead to less efficient road capacity usage and traffic jams.

Use case Parking spot availability

- *Reserve parking spots for own benefit.* The attacker can claim less free parking spots are available in a certain area than there actually are. If success-

ful, the attacker is the only one that knows the physical location of the actually free parking spots and can use them.

— *Falsify parking spot sums to influence inner city traffic behavior.* Attackers can report increased numbers of free parking spots in one region and low numbers of free spots in other regions to influence drivers to drive to the region with the most free parking spots. This can lead to traffic congestions.

— *Falsify availability information of specific parking lots for economic damage.* Influencing the availability information of specific areas can lead to lower usage of the parking spots in question. Consequently, the operators of these parking spots may loose money in terms of parking fees.

4.4.3 *Exemplary attackers*

While many different specific attacks on in-network aggregation are conceivable, the basic goal of an attacker can be characterized as aiming to make the aggregated view of the real world deviate as much as possible from the actual situation (cf. Section 4.4.2). In all cases, the attacker can be assumed to be an insider attacker, as argued in Section 4.4.1. We therefore introduce the following exemplary attacker profiles, which cover a wide range of specific attacks and will be used, together with Definition 11, to evaluate existing and proposed security mechanisms.

— The SENSOR attacker creates false atomic observations, but does not manipulate the decision or fusion functions. For example, the attacker could report velocity values of 0 km/h whereas the actual values are much higher. The attack is relatively easy to implement, because the attacker can target either the vehicle's on-board sensors or the software running in the vehicle, as discussed in Section 4.4.1. Moreover, the SENSOR attacker does not require specific knowledge about the aggregation scheme. On the other hand, the SENSOR attacker's influence is likely low even if an aggregation scheme is not specifically protected against attacks, as explained in Section 4.3 and formalized in Equation (4.7).

— The FAKE attacker aims to convey a situation that is not existent in the real world by altering aggregates in addition to observations. For example, the attacker could create aggregates about a presumed traffic jam whereas vehicles are actually moving at higher speeds in the claimed geographic area. The attack is harder to implement than the SENSOR attack, but the potential impact is higher, as explained in Section 4.3 and formalized in Equation (4.6).

— The CONCEAL attacker aims to conceal a situation that actually exists in the real world. For example, the attacker could create aggregates that claim free-moving traffic in a certain region, whereas there is actually a traffic jam in said region.

The FAKE and CONCEAL attackers are very similar at first sight, but we have choseen them as separate attacker profiles for two reasons. First, some aggregation schemes we discussed in Section 3.5 distinguish between a "regular" situation and an "irregular" situation to save dissemination bandwidth. For example, free-flowing traffic would be considered "regular" and a traffic jam would be considered "irregular." Consequently, only "irregular" information is disseminated and the absence of information is interpreted to mean the situation is "regular." Such aggregation mechanisms make the FAKE attack easier than the CONCEAL attacker. In case of the FAKE attacker, benign vehicles would not disseminate any information; the attacker's false warnings are, therefore, the only available information to other vehicles. In absence of conflicting information, benign vehicles would likely believe the attacker's aggregates. The situation for the CONCEAL attacker is the opposite: the attacker would need to suppress information from almost all benign vehicles to be successful, which might be harder than disseminating their own information.

Second, even aggregation mechanisms that do not distinguish between "regular" and "irregular" information often prioritize information about "irregular" events [e. g., 3]. For instance, information about traffic jams might be forwarded with higher priority than information about regular traffic. As a result, FAKE is again easier than CONCEAL, because the fake "irregular" information could be prioritized.

4.5 THREATS AND VULNERABILITIES

Attackers can use different techniques to achieve the aforementioned attack goals. Those techniques can be mapped to the components of the generic aggregation architecture presented in Section 3.6. Note that it is not necessary for attackers to adhere to or implement the aggregation scheme as presented for the categorization to work. The essence of the categorization lies in the strategies that are employed to attack the specific components.

4.5.1 *Local Observations*

Local observations are the source of all information that is disseminated by an aggregation scheme. Because an attacker possesses one or more valid key pairs, they can create new observations with arbitrary content. Given an honest majority, attacks that only focus on false atomic observations do not have a high impact, because outlier detection strategies can be employed to detect false local observations [e. g., 138]. However, if values of single observations are not bound, that is, negative speeds or negative parking spot numbers are allowed to be reported, unusually high temperatures are allowed, and so forth, even fake local observations can have a high impact on aggregation results. In that case, attackers can craft a fake local observation to significantly skew aggregate content to meet the attacker's goal. We will elaborate more on this threat in Section 4.5.3.

4.5.2 *Decision Component*

An attack targeting the decision component means that an attacker selects a set of valid aggregates or observations that is beneficial for the attacker goal and combines them to a new aggregate. The fusion functions are applied correctly in this case. Only the aggregation decision is manipulated. Suppose an attacker wants to fake free flowing traffic on a stretch of congested road. A possible strategy is to aggregate observations from the surrounding stretches of free-flowing traffic, claiming that the whole region between these observations is also free-flowing traffic. Similarly, weather reports or parking spot information can be combined.

Attacker success depends on information types.

Note that the success of this attack depends on the type of information that is aggregated. We need to distinguish two possible types: (1) information represents binary events, and information dissemination is only triggered if an event occurs; (2) information is real-valued and represents the state of the real world; events are only interpreted by the receivers of information. An example for (1) is the dissemination of binary information about traffic jams. If only aggregated traffic jams are communicated, an attacker cannot hide a possible traffic jam, because no information is available from areas where no traffic jam is present. However, an attacker can craft longer traffic jams by combining nearby aggregates of two disjoint traffic jam situations. An example for (2) is the dissemination of average speed information. Here, values can be influenced in both directions, because reports about both free flowing traffic, as well as traffic jams, are available.

In both cases, no information contained in the existing valid observations needs to be manipulated for the attack. Only the decision that the values are combined is changed. Such attacks are easily detectable in simple schemes based on fixed-size segments. However, if the decision whether to combine two aggregates depends on multiple context factors, it can be difficult for receiving nodes to detect the wrongfully combined aggregates.

4.5.3 *Fusion Component*

An attack on the fusion component means that the way in which information is combined is manipulated. That is, either a calculation error is introduced to the actual fusion function, or the result is replaced by a fixed value. For example, an attacker can use a set of aggregates about a certain area as input and aggregate them further, but replace the contained values by a different situation.

A variant of this attack stems from an attacker being able to introduce own valid observations into the network. This can be exploited in the following way. The output of many fusion functions can be arbitrarily modified by a single input value if the input values are not bounded. For example, the average of a range of values $v_1, \ldots, v_n$ can be manipulated to be a by choosing a single additional input value as $a \cdot (n+1) - v_1 - \cdots - v_n$. Similarly, the minimum and maximum can be influenced to be arbitrarily small and large, respectively.

In this case, neither the decision nor the fusion components are manipulated. An attacker is simply exploiting that it can contribute valid own contributions.

4.5.4 *World Model*

Attacking the world model means that an attacker produces arbitrary aggregates and adds them to the local world model for further dissemination. This represents forging information without taking into consideration any existing aggregates or observations that might support the attacker view. Although this seems to be a simplistic attack at first sight, it can be successful against an otherwise unsecured aggregation scheme. The enabling factor here is the lossy aggregation. For the receiver of aggregates, it is unclear what the history of said information is beyond the direct neighbor node. Thus, an attacker can claim to possess arbitrary values from arbitrary regions of the network.

4.5.5 *Dissemination*

Vulnerabilities of the dissemination component characterize all forms of packet dropping attacks. A null dissemination function means that an attacker does not forward information at all, for instance, to disrupt the service of the system. Likewise, an attacker can disseminate only those aggregates further that support their goal. While not as severe as wrong information, the absence of up to date correct information can still lead to wrong application choices. This is especially true due to the temporal nature of the disseminated information.

Suppose for instance a wet stretch of road is just freezing over, and an attacker wants to hide the dangerous road conditions. Nodes that passed the stretch while it was still wet but comparatively safe will have disseminated said information. Because it is beneficial for the goal of hiding the black ice, the attacker forwarded the aggregates. But as soon as other nodes collect information about the newly frozen road, the attacker filters them out and still disseminates the older information about the wet road. In lack of fresher information, attacked nodes will continue to assume the upcoming road is just wet, and they will not be aware of the black ice until a threshold for the maximum age of relevant information is reached.

4.6 SECURITY APPROACHES

During the previous sections of this chapter, we have already introduced intuitive notions of different security approaches. For instance, in Section 4.3.2, we have discussed semantics of cryptographic signatures and their possible combination with physical models to achieve integrity against insider attackers. These are just two examples for different approaches to implement secure in-network aggregation mechanisms. In general, we can distinguish four types of basic approaches for security, which offer different trade-offs between resilience against attackers – as discussed in Section 4.3.2 – and requirements for efficient aggre-

gation outlined in Section 3.4. Usually these approaches are not used in isolation but are combined to create a secure aggregation scheme. In the following, we will provide an overview of each category, introduce primitives and tools that are commonly used for achieving security with those approaches, and discus benefits and drawbacks of each category and tool. Combinations of the approaches we discuss here are the basis of our security mechanisms we present in Chapters 7 to 10.

4.6.1 *Cryptographic mechanisms*

Cryptographic mechanisms use cryptographic measures – typically digital signatures – to protect the integrity of information. The basic idea here is that a signature conveys a certain confidence of the signature's creator in the information that is signed, as discussed in Section 4.3.2. The signer becomes a "witness" for the information contained in messages. For atomic observations, this means that the signer claims to have directly observed the contained information using local sensors. For aggregates, the connotation of a signature is usually that the signer claims to have correctly performed aggregation decision and fusion and only used available information in the world model.

Signatures also identify the signer. For a fixed (short) time period, a signer can only use a very limited number of signing keys in parallel (see Section 4.2), therefore the influence is restricted, even if signing keys do not allow to reveal the signer's identity and even if key ids change over time. These attributes of signatures are often used in combination with physical models or redundancy checks (see Section 4.6.3). For example, a majority voting scheme combines signatures with a model of the likelihood that an attacker can influence the majority of voters.

For in-network aggregation, a main drawback of cryptographic signatures is their required overhead. As discussed in Section 4.2, a single ECDSA 256 bit signature including public key and certificate is 221 bytes. A naïve approach that attaches all original witness signatures for an aggregate that covers a large geographic region would thus require several kilobytes of data. Given a maximum transmission unit (MTU) of 1500 bytes as specified in IEEE 802.11 [210], at most 6 signatures could be accommodated without fragmentation, even when disregarding header fields and other payload. Without the original witness signatures, however, receiving vehicles cannot verify whether the aggregating vehicle adhered to the aggregation protocol.

Gentry and Ramzan [189], as well as others [e. g., 32, 73, 172], coined the term *who signed what* to describe the minimum information necessary for verification.

> For any signature scheme, one must have a description K of *who signed what*. […] K has a certain Kolmogorov complexity [102], which is the minimum number of bits that are needed to convey the value of K. Therefore, […] the ultimate objective is to find a

> signature scheme that is as close as possible to being "Kolmogorov-optimal." [189]

Applied to in-network aggregation, *who signed what* can be understood as the proof that a number of vehicles *(who)* attest to the correctness of – possibly aggregated – messages *(what)*. The goal is to find the smallest (i. e., *Kolmogorov-optimal*) representation of this description. The naïve implementation discussed above requires a description of size $K = 221 \cdot n$ for n witness vehicles. Multi-signatures, aggregate signatures, and identity-based signatures aim to lower the size of K using different assumptions about signed messages, signing parties, and changes in PKI infrastructure. In the following, we will give an overview of these cryptographic approaches.

Multi-signatures.

A multi-signature σ_M is a combined signature of n signers on the same message m [31]. Written multiplicatively,

$$\sigma_M(m) = \prod_{i=1}^{n} \sigma_{sk_i}(m). \tag{4.8}$$

Here, all signatures and messages are elements of a Gap Diffie-Hellman (GDH) group where the decisional Diffie-Hellman problem (DDH) is easy, but the computational Diffie-Hellman problem (CDH) is hard. Certain elliptic curves together with bilinear pairings have been shown to give rise to suitable GDH groups [35].

See, for instance, Boneh et al. [35] for an explanation of GDH, DDH, and CDH.

The verifier can check the multi-signature given

$$(m, \sigma_M(m), pk_1, \ldots, pk_n, cert_1, \ldots, cert_n). \tag{4.9}$$

The size of σ_M is considerably smaller than the concatenated size of all original signatures. Applied to in-network aggregation, multi-signatures require all signers to agree on a common message (i. e., aggregate) before initiating the signing process. This requirement can be problematic in highly mobile scenarios. Vehicles may only be in mutual communication range for a short period of time, and they may not be able to engage in an agreement protocol. Moreover, a list of all public keys and certificates is still required for verification; only the signature size is constant, independent of the number of signers.

Aggregate signatures.

An aggregate signature σ_A is a combined signature of n signers on n messages $m_1, \ldots, m_n$ [34]. Written multiplicatively,

$$\sigma_A(m) = \prod_{i=1}^{n} \sigma_{sk_i}(m_i). \tag{4.10}$$

The verifier can check the aggregate-signature given

$$(m_1, \ldots, m_n, \sigma_A(m), pk_1, \ldots, pk_n, cert_1, \ldots, cert_n). \tag{4.11}$$

Again σ_A's size is considerably smaller than that of the concatenated original signatures, but a list of all public keys and certificates is needed for verification. Moreover, all original messages need to be known. Since fusion functions used

for aggregation are often not invertible (cf. Section 3.6.3), these original messages can often not be reconstructed.

Identity-based signatures and homomorphic signatures.

To further optimize size, identity-based signatures [47], which are signature schemes based on identity-based encryption schemes, have been investigated [e. g., 14]. These reduce the size of public keys by a constant factor compared to regular ECDSA signatures but require extra computational power in vehicles. Moreover, recent research on homomorphic cryptography [53, 207] is promising for application in in-network aggregation. Essentially, the goal of homomorphic signatures is to allow a pre-defined set of operations, such as calculating sums and averages, on signed data while maintaining a valid signature on the result of those calculations. While homomorphic signatures have been applied to other information dissemination protocols (e. g., network coding [40, 91, 178]), their application to practical aggregation schemes is still an open issue. A basic hindering factor is that homomorphic encryption typically allows each entity that modifies signed information a certain set of operations. To be applicable to aggregation, the modifying entities would need to be restricted to the correct use of aggregation decision and fusion functions, but any incorrect use should invalidate the (homomorphic) signatures. Such fine-grained restrictions are currently not implemented by homomorphic signature schemes. In addition, most homomorphic signature schemes assume that all original values are signed by the same party. For instance, Boneh and Freeman [33] present a homomorphic signature scheme that can derive signed statistical values such as averages and standard deviation from a signed set of inputs, but each input value needs to be signed with the same secret key.

Summarizing, cryptographic protection approaches are useful, because they restrict attacker influence and enable other mechanisms, such as majority voting schemes and other data consistency mechanisms. However, the use of signatures is costly: their size is inhibitive for sole use to protect in-network aggregation schemes, even when optimizations such as multi-signatures and identity-based cryptography are employed.

4.6.2 *Interactive mechanisms*

Interactive mechanisms are approaches where vehicles engage in an interactive protocol, involving multiple message exchanges, to evaluate correctness of information. Two basic forms have been investigated for use in aggregation protocols. We call these approaches *agree-then-commit* and *commit-then-verify.* Both relate to a generic security called *aggregate-commit-prove* approach originally proposed by Przydatek et al. [142] for WSNs.

See Section 5.2 for a discussion of Przydatek et al.'s work.

The first category, that is, *agree-then-commit,* is necessary, because many cryptographic protection primitives, such as the multi-signatures discussed in Section 4.6.1, require a common agreed-on value before signatures can be generated. To reach agreement, vehicles first form a temporary stable structure (e. g., a cluster [4, 145]). Then, vehicles within the structure agree on a common value and broadcast it to all other vehicles. Finally, vehicles can attach their

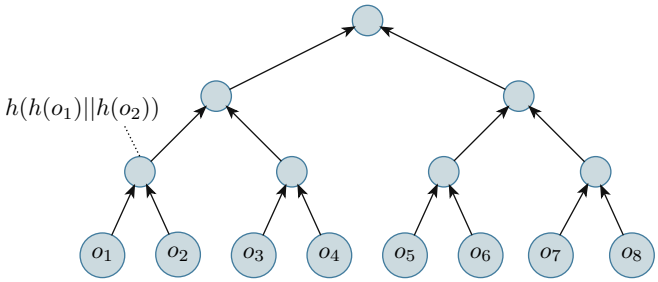


Figure 4.4: Schematic of Merkle hash tree calculation for 8 observations $o_1, \ldots, o_8$. At each node, a hash of both child nodes' concatenated hash values is calculated, as exemplarily shown for o_1 and o_2's parent node.

signature to the previously-agreed value to act as witness for the correctly performed aggregation. Often, the created structures correlate to single-hop communication range. Vehicles can then check the agreed aggregate value against their own sensor readings, employing a form of physical model as additional security mechanism (cf. Section 4.6.3). Drawbacks of the *agree-then-commit* approach are that stable structures might be hard to maintain during the required message exchange due to vehicle mobility and that the additional consistency checks require restricting the aggregation area to the single-hop communication range.

The second category, *commit-then-verify,* is based on the idea of commitment schemes [74], and has been applied to aggregation by Picconi et al. [141]. A generic commitment protocol for in-network aggregation can be described as follows. An aggregating vehicle (a) uses observations $o_1, \ldots, o_n$ to calculate an aggregate $A = (o_1 \bowtie \ldots \bowtie o_n)$. We assume that the aggregator also possesses the observation signatures from the original senders: $\sigma_{sk_1}(o_1), \ldots, \sigma_{sk_n}(o_n)$. The aggregator creates a commitment on the used observations using a Merkle tree [225]:

$$MT(o_1, \ldots, o_n) = h(\ldots h(h(o_1)||h(o_2))|| \ldots ||h(h(o_{n-1})||h(o_n)) \ldots). \tag{4.12}$$

Here, h is a cryptographic hash function and $||$ denotes concatenation of values. Figure 4.4 shows a schematic representation of the hash tree calculation. First, hashes are calculated for all leaf nodes. At each higher level, a hash is calculated over the concatenated child node hashes. Finally, only a single hash value is calculated at the root node, which represent a commitment to all leaf values. The aggregator then sends $(A, MT(o_1, \ldots, o_n), \sigma_a(A, MT(\cdot)))$ to the neighboring vehicles. To verify whether the aggregator performed the aggregation correctly, receiving vehicles randomly choose a number $k < n$ of observations that the aggregator should reveal. The aggregator must then present the k original observations, their original signatures, and a number of additional signatures to verify whether the presented values were used in the Merkle tree hash

MT. Moreover, the verifier can check whether the values in the original observations are consistent with the aggregated value. The value of k can be adapted to achieve a trade-off between security and bandwidth overhead.

A problem of the *commit-then-verify* approach is that the aggregator who committed to an aggregate may drive out of communication reach directly after the aggregate was transmitted. In a more complex aggregation setting, the approach may not work at all, because aggregates are generated gradually and hierarchically and there is not one specific aggregator that creates the final aggregate from a set of observations.

In summary, interactive security mechanisms can offer a good trade-off between bandwidth usage and achieved security, and they can enable a number of data consistency checks, such as comparison with local sensor values after agreement, when combined with cryptographic security mechanisms. But their requirement for temporary stable groups of vehicles can be hard to achieve in highly mobile settings.

4.6.3 *Data consistency mechanisms*

Data consistency mechanisms compare multiple items of information with each other or with off-band sources of information to detect or filter outliers. As evident from previous discussions, cryptographic mechanisms and interactive mechanisms are usually combined with data consistency mechanisms to add semantic meaning and interpretation to values. Moreover, data-centric mechanisms are a promising approach for aggregation security, because they often do not require extra overhead. Besides in-network aggregation, data-consistency checking has already been successfully applied to beaconing and other message dissemination mechanisms [e. g., 90, 109, 18]. Simple forms of data plausibility checking are already foreseen by standardization bodies [192]. For in-network aggregation, three types of data consistency checks can be applied: *alignment with local sensors, physical models,* and *data redundancy.* All three categories can be combined with each other for more complex data consistency mechanisms.

Alignment with local sensors.

The *alignment with local sensors* approach uses local sensors as a trustworthy source for verification of received aggregates. For example, if the own velocity indicates a traffic jam, and the vehicle receives aggregates claiming free-flowing traffic in its geographic area, the aggregate can be discarded because of the more trustworthy local sensors. Likewise, Lidar, radar, or other scanners could be used to match claimed free parking spot indications with local sensor observations by sensing parking spot occupancy. Alignment with local sensors requires that the verifying vehicle is itself in the aggregate's claimed geographic area. It can be combined with cryptographic approaches to convey the performed verification to further away vehicles, for instance, using the agree-then-commit scheme described in Section 4.6.2. Without adding extra cryptography, we can argue that if the majority of vehicles is honest, and if each vehicle checks aggregates against local sensors if it has corresponding local sensor readings, and

if each vehicle drops suspicious aggregates, the suspicious aggregates should never be disseminated to further vehicles. But the success of this approach depends on the attacker's position in the network and on whether aggregates pass through vehicles located in the geographic area they have information about before they reach further vehicles.

Physical models.

Physical models are almost always used to give semantic meaning to values in data consistency mechanisms. Some examples for physical models, ranging from simple to more complex, are:

1. "a vehicle never drives faster than 300 km/h;"
2. "all vehicles drive on (or very close to) the road network;"
3. "an event that extends along m meters of road is observed by at least $f(m, d)$ vehicles, where f is a linear function and d the vehicle density on the road;" and
4. "if v_1 and v_2 are velocities measured on the same stretch of road then $|v_2 - v_1|$ can only be large if the positions these velocities were measured are further apart."

For traffic information systems, a wide range of physical models have been developed, which model behavior on a microscopic level [e. g., 128] or macroscopic level [e. g., 80, 133]. A basic problem of all physical models is that they are derived from "regular" behavior. But as long as behavior is "regular," the informational value of messages reporting this behavior is limited. It is messages about "irregular" behavior that are potentially most useful. For instance, a vehicle that is *not* close to the road network (Example 2) or two vehicles close-by that *have* very different velocities (Example 4) could indicate an accident that *should* be warned about. Still, models can serve as a useful baseline especially for securing in-network aggregation. Because aggregation mechanisms are not directly safety-related (cf. Section 2.2), we can assume that other protocols with corresponding security mechanisms are in place to detect and warn about irregular behavior. Such safety-focussed integrity protection mechanisms are, however, outside the scope of this thesis.

Data redundancy.

Finally, *data redundancy* mechanisms assume that reports about large events will likely be received via multiple dissemination paths. In-network aggregation protocols often use opportunistic dissemination and disseminate the same information multiple times, which supports the hypothesis that sufficient redundancy exists. We formalized [10, 11] the impact of redundancy on attacker detection and analyzed the level of redundancy for different dissemination protocols. We will discuss results of our redundancy analysis as part of our redundancy-based security mechanism in Chapter 10.

In summary, data consistency mechanisms are an important building block to interpret the semantics of information and use those semantics to detect attacks. Purely data-centric mechanisms have the benefit that they use little or no extra bandwidth. To analyze data consistency of information originating from further away regions, consistency needs to be combined with cryptographic

mechanisms so that attackers with certain positions in the information flow graph cannot wrongly influence aggregation results. Proper combination of data consistency and cryptography mechanisms will be one of the main contributions in this thesis, which we discuss in Chapters 6 to 11.

4.6.4 *Trusted-hardware-based mechanisms*

Trusted-hardware-based mechanisms use trusted hardware components to render attacks on aggregates or observations impossible or at least infeasible. These trusted components may be used establish a trusted perimeter in the vehicle where software and sensor values cannot be tampered with. Different protection levels are conceivable [171]. In the simplest case, only cryptographic keys can be stored in an HSM. This is the approach currently foreseen by standardization [192, 193]. In addition, vehicle sensors can be tamper-protected or even the whole software stack running inside the vehicle. We significantly contributed to a generic trusted-computing-based solution that aims to protect data consistency in cITS by allowing trusted and controlled execution of data processing components [15, 17]. Here, all custom applications run within a controlled application environment [8], which mediates access to sensor values, as well as network access. Although originally developed for privacy protection, the developed architecture could be adapted and extended to achieve integrity protection for in-network aggregation. Given the strict cost requirements for vehicular development, it is, however, unlikely that the full software stack in vehicles will run inside a trusted hardware platform. We therefore assume only protection of key material as discussed in Section 4.4.1 and will focus on cryptography-based, interactive, and data-consistency-based protection mechanisms in Chapters 7 to 10.

4.7 CHALLENGES

During the security analysis of in-network aggregation, we have already identified a number of challenges for securing aggregation. Section 4.3.2 has differentiated resilience for in-network aggregation from other non-aggregating mechanisms from a security goal perspective. Likewise, numerous examples in Section 4.6 have shown security challenges that are unique to aggregation. In the following, we will summarize the main challenges for resilient in-network aggregation based on the security analysis.

See also Sections 4.3.2, 4.4, 4.5.1.

COLLABORATIVE PARTICIPATION In an aggregation protocol, each vehicle contributes to the goal to estimate the real situation in the surroundings of a road network. Not only does each vehicle contribute its own sensor readings to the overall result, but each vehicle also performs decision and fusion operations on received values before further dissemination. Therefore, attackers can modify their own contributions, but they can also modify or make up aggregated information. A security mecha-

nism needs to verify both observations and aggregates to be successful in detecting attackers.

See also Sections 3.4.3, 3.6.3, 4.6.1.

LOSSY DATA COMPRESSION In order to scale to large geographic areas, aggregation must use the semantics of information for compression. As a result, the original observations cannot be reconstructed from aggregated values. Such lossy data compression removes the redundancy and subtle diversity in original observations that could otherwise be used to detect outliers or other spurious information. Moreover, lossy compression invalidates cryptographic signatures on the original observations.

See also Sections 2.3, 2.4.

LACK OF CENTRAL ENTITIES In a completely dynamic aggregation scheme, no vehicles perform special roles during aggregation and no RSUs or backend systems perform designated tasks. In other domains, like WSNs (cf. [22]), central entities are often assumed to have a higher trust level and are more protected against attackers. These trusted central entities can then validate observations, perform aggregation, and disseminate the result. For this to work, central entities need to be constantly connected to the network. In VANETs, central entities may not always be available, and secure aggregation mechanisms need to work on a peer-to-peer basis.

See also Sections 2.3, 4.6.2.

EPHEMERAL NODE CONTACT Vehicles may meet with high relative speed. The sender of a message likely moved out of mutual reception range directly after disseminating a message. Static groups of vehicles only form automatically in traffic jam situations. In other situations, clusters of vehicles have to be detected using a designated protocol. Without the possibility to inquire senders about their aggregation, for instance, using a commitment protocol and probabilistic checks, all information necessary to validate aggregates needs to be attached directly. Likewise, agreement protocols suffer from ephemeral contact, because they require multiple communication interactions to agree on a common view between vehicles.

See also Sections 2.5, 3.4.1.

SCALABILITY REQUIREMENTS In order to disseminate information in large geographic areas, the granularity of its representation must be *reduced* quadratically [152]. However, for most cryptography-based approaches, the amount of security overhead *increases* linearly in the covered geographic area. Whenever security overhead is removed, the likelihood of a successful attack increases.

Together, these challenges make designing security mechanisms for in-network aggregation a uniquely hard problem. Solutions need to be flexible and offer parameters to find a trade-off between scalability and security requirements.

Table 4.1: Overview of attacker attributes

	SENSOR	FAKE/CONCEAL
Capabilities	Manipulate sensors.	Manipulate software or extract keys.
Goals	Influence navigation decisions.	Influence navigation decisions.
Exploited vulnerabilities		
Local observations	Create false reports.	—
Decision	Possibly merge false reports.	Possibly discard correct aggregates, keep false aggregates.
Fusion	Possibly merge false reports.	—
Dissemination	—	Disseminate false aggregates.
World model	Influence traffic representation.	Influence traffic representation.

4.8 SUMMARY

In current VANET research, security and privacy considerations have been a major part of research activities early on. Current standardization foresees protection mechanisms to safeguard message integrity against outsider attackers while maintaining a minimum level of driver privacy. Intuitively, in-network aggregation hinders security while it fosters privacy. The main asset to protect is information integrity. In-network aggregation collects information about real-world phenomena, and this merged and abstracted information should represent the actual situation as closely as possible.

However, insider attackers, which we constitute a conceivable and realistic scenario, can alter both own observations about the real world and the aggregated view. Especially the latter can negatively influence information correctness. The former, modification of own observations, exemplified by the SENSOR attacker, is negligible if attackers only control a fraction of all vehicles. We therefore focus our security goal, formalized in Definition 11, on preventing attacks on aggregated values. Two exemplary attackers concretize this goal and will be used to evaluate our security mechanisms: a FAKE attacker trying to create inexistent situations and a CONCEAL attacker aiming to hide actually existing situations. Table 4.1 shows an overview of all representative attackers and their main attributes. The challenges in securing in-network aggregation are due to the algorithms' highly collaborative, distributed, and decentralized operation in a mobile network with ephemeral node contact and tight scalability requirements. Next, we discuss a number of existing proposals for in-network aggregation to identify promising approaches and shortcomings.

RELATED WORK

5.1 OVERVIEW

5

Parts of this chapter are a revised and extended version of our related work survey's secure aggregation discussion [2, Sec. IV.D].

A number of research works exist that investigate resilient aggregation in different domains. We will review the most relevant works that present secure aggregation mechanisms for VANETs, as well as mechanisms from two related fields.

WIRELESS SENSOR NETWORKS Before the advent of VANETs, WSNs have been the predominant application area for aggregation mechanisms, as we mentioned in Section 3.5.1. Many security mechanisms for in-network aggregation in VANETs were inspired by schemes originally proposed for WSNs. Because network characteristics in VANETs are largely different, however, we only briefly review seminal work in WSNs in Section 5.2.

EVENT VALIDATION Mechanisms for secure event validation in VANETs do not perform in-network aggregation in the strict sense of Definition 1. Here, an event is assumed to be detected by some means and then validated by signatures from a number of witnesses. Events are binary (e. g., "an accident occurred at position x"), and the event description is usually created once and not altered afterwards.

We review event validation mechanisms in Section 5.3, because some secure aggregation mechanisms adapt and extend their concepts to apply them to more complex information. Section 5.4 discusses work on securing in-network aggregation in VANETs, which is most related to the mechanisms proposed in this thesis.

5.2 WIRELESS SENSOR NETWORKS

WSNs are ad hoc networks formed by large numbers of small nodes equipped with sensors, as shown in Figure 5.1. These nodes are often deployed in uncontrolled environments, but the nodes themselves are owned and controlled by a single entity [20]. Contrary to vehicles, sensor nodes do not move at all or move only slowly, for instance, if used for wildlife monitoring [162]. Moreover, only few entities – or even just a single entity – collect the sensed values. With these properties, WSN make it easy to assign designated routing roles to a selected subset of nodes, introducing a hierarchy. At the lowest level, sensor nodes only collect and forward information. Designated aggregator nodes perform aggregation and forward the aggregated information further towards the sink. The stable network structure is especially beneficial for security [150], because it allows to run interactive agreement protocols amongst nodes. And stable positions allow to make assumptions about higher trustworthiness of designated nodes if they are deployed in secured areas that are hard to access for

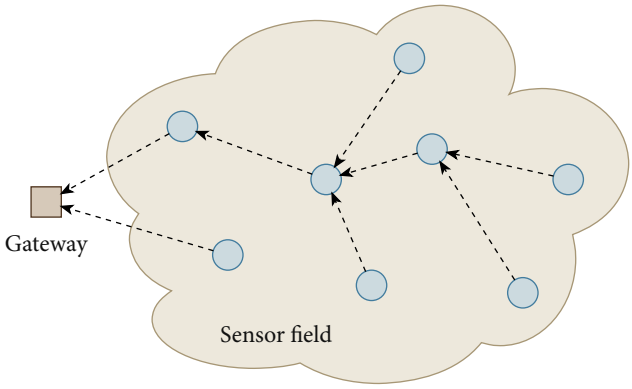


Figure 5.1: Example WSN: a number of wireless sensor nodes establish an ad hoc network and forward information towards a sink. Often, a tree topology is established for forwarding, as indicated by the dashed arrows.

attackers. A second distinguishing factor is the optimization of WSNs towards energy usage. Sensor nodes are typically equipped with small batteries [21, 155, 174], which necessitates the design of energy-preserving protocols. Energy constraints also limit the use of cryptography. For instance, many protocols [e. g., 150] use symmetric cryptography rather than public key cryptography to secure WSN aggregation. In VANETs, energy is not a major limiting factor, which enables to consider a broader range of cryptographic mechanisms, such as those discussed in Section 4.6.

Many approaches for secure aggregation in WSNs exist, and we refer to Alzaid et al. [22] and Sang et al. [150] for a more extensive survey discussion of secure aggregation in WSNs. Due to the different network characteristics of the WSN domain, we discuss exemplary data-centric and cryptography-based secure aggregation approaches in the following rather than providing an extensive overview. And we use these exemplary protocols to discuss which ideas may be transferred to VANETs, and which ideas are specific to WSNs.

Przydatek et al. [142] introduce the problem of secure information aggregation in WSNs and present first integrity protection schemes based on symmetric cryptography. In their paper, the term *stealthy attacks* is coined for "attacks, where the attacker's goal is to make the home server [i. e., the sink] accept false aggregation results, which are significantly different from the true results determined by the measured values, while not being detected by the home server." They further propose the generic approach *aggregate-commit-prove* to achieve data integrity in sensor networks. First, the aggregator collects information from sensors and computes the aggregation result. Second, the aggregator transmits the result to the sink together with a commitment – for instance, using Merkle trees [225] – on the values used. Finally, the sink engages in an interactive protocol with the aggregator to reveal a subset of the original values used in the aggregation.

Mahimkar and Rappaport [117] propose an extension of Przydatek et al.'s scheme. Nodes are grouped into clusters of close-by nodes, and cluster heads perform the actual aggregation. First, the cluster heads collect sensor readings from each node in their cluster, aggregate them, and disseminate the result. The cluster member nodes compare the average against their own value, reject it if the values differ by more than a pre-defined threshold, and finally sign the average using a share of a threshold signature key [134]. The aggregator then computes a complete signature using the received partial signatures and forwards the result to the sink. An attacker cannot compromise the aggregated result unless he controls enough nodes to create a valid complete signature or compromises an aggregator node. The cluster head then sends the aggregation result to the sink. Moreover, the cluster head commits to the result using a Merkle tree as in Przydatek et al.'s proposal.

Intuitively, a threshold signature system distributes key shares to all participants. Each participant can create a partial signature, but only the combination of at least t shares can create a valid complete signature where t is a fixed threshold.

Labraoui et al. [105, 106] presents a monitoring-based approach for secure aggregation. They assume a network with designated aggregators that act as cluster heads and aggregate information of surrounding nodes. Moreover, aggregation is initiated by the sink, which sends a query for a required value. During the query phase, randomly selected nodes are designated as monitor nodes. Monitor nodes passively overhear the aggregation process. Whenever an aggregator node tries to influence the aggregation results by disseminating false aggregates, the monitors will send an alert, and aggregators can be evicted from the network. The approach is specific to WSNs, because the scheme's security partly relies on the centralized sink to select random monitor nodes. Also, nodes cannot move rapidly, because monitors need to overhear all broadcast communication between other nodes and aggregators to detect possible attacks.

Castelluccia et al. [44] discuss a cryptography-based approach for secure aggregation. In contrast to the previous schemes, the focus is on confidentiality. The goal is that a passive attacker cannot extract information from the aggregation process. To achieve confidentiality, the authors use homomorphic cryptography [207] to calculate averages and variances of values. The advantage of homomorphic encryption is that aggregators do not need to decrypt sensor values. In terms of integrity, it can be argued that aggregators cannot easily influence the aggregation result if they do not possess the necessary decryption keys. They can, however, disrupt aggregation by arbitrarily changing encrypted data.

Rather than introducing a specific secure aggregation scheme, Bhaskar et al. [28] analyze fundamental trade-offs between achieved security and bandwidth overhead for a specific case of WSN aggregation. The authors assume a number of sensor nodes that aggregate information using an arbitrary hierarchy. The final aggregation result is forwarded to the sink, here called verifier. Each node contributes a value within a fixed range $[0, R]$, and aggregation is performed by summing up values. Each node shares a secret key with the verifier, which is used to create message authentication codes for each sensor value. The attacker can compromise a subset u of the sensor nodes, and the attacker's goal is to

Message authentication codes are the symmetrical equivalent of cryptographic signatures; they can be created using a keyed hash function [222].

influence the aggregation significantly, that is by more than $u \cdot R$. For this setting, the authors prove that a single-round aggregation protocol needs to have a bandwidth overhead in the order of $\Omega(n)$ where n is the number of nodes, or it will not detect the attack. Intuitively, this minimum bandwidth overhead is as least as high as that of a similar information collection protocol that uses no aggregation at all. Even though the paper is limited to a very specific setting of aggregation, it provides the first formal impossibility result for truly resilient aggregation in a domain related to VANETs. Applied to the resilience definition we introduced in Section 4.3.2, the result would mean that resilience as stated in Definition 13 can only be achieved with security overhead that is linear in the number of contributors to an aggregate. While we cannot directly apply Bhaskar et al.'s result to VANETs due to the different aggregation setting, the result still suggests that secure in-network aggregation mechanisms for VANETs may need to make trade-offs between security and bandwidth overhead.

Here $\Omega(\cdot)$ denotes the Big Omega notation, which states a lower bound on the required overhead [101].

5.3 EVENT VALIDATION

Petit et al. [137] present an event validation mechanism for VANET safety messages, which they later extend to support dynamic thresholds [138] and provide an overhead evaluation [140]. To validate event reports, vehicles collect signed warnings about the same event from different vehicles. A warning for the driver is created only if the number of warnings received passes a threshold. Rather than choosing a static threshold, the authors propose to use a dynamic threshold. The dynamic threshold calculation factors in driver reacting time, breaking distance required to avoid an accident, current speed of the own vehicle, and the severeness of events. Thereby, the authors achieve a trade-off between security requirements and the time-critical nature of safety warnings.

A voting mechanism to prove emergency events is used by Zhu et al. [182]. Similar to Petit et al.'s work, the authors collect a number of signatures on the same event before it is considered valid. But in Zhu et al.'s scheme, aggregate signatures [35] are used. The advantage of aggregate signatures is that they can be combined into a single signature during dissemination. Disseminating only a single signature saves bandwidth and enables so-called batch verification [43] at the receiver: only one (aggregate) signature needs to be verified to ensure that all vehicles' signatures that were used to generate the aggregate signature are correct. Viejo et al. [164] discuss a variation of the scheme, which also uses aggregate signatures for efficiency.

See Section 4.6.1 for an explanation of aggregate signatures.

Garofalakis et al. [72] offer a straightforward security mechanism to event validation using secured FM sketches. The authors assume binary events that all vehicles can either claim to witness or not. Moreover, it is assumed that an upper bound for the total number of vehicles in the network is known. For each bit set to 1 in a sketch, a proof is kept that contains the node ID that set the bit to 1, the bit position in the sketch, the vehicle's atomic observation (i. e., sensor values), and the vehicle's signature on these values. This approach protects against inflation of the FM sketch value to the extent that an attacker needs to

FM sketches are a probabilistic counting structure; for an introduction see Section 3.5.

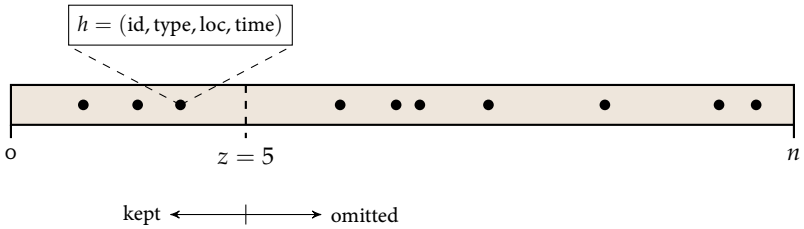


Figure 5.2: Structure of a secured z-smallest approximation.

provide a valid signature from different vehicles for each bit set to 1. Deflation protection is achieved by adding a second FM sketch that counts the complement value, that is, $N - v$ where N is the expected upper bound of the value and v is the value to be counted.

Hsiao et al. [82] propose to use z-smallest probabilistic counting instead of FM sketches to proof the occurrence of events. The idea of z-smallest is that, given n elements uniformly distributed between 0 and 1, the z-smallest element gives an approximation of n by calculating z/c where c is the value of the z-smallest element, as shown in Figure 5.2. To protect against inflation, each vehicle signs a hash of its vehicle id, the event type, location segment, and time. Only the z-smallest signatures are kept with the aggregate. The idea is that an attacker cannot produce enough signatures on hashes that fall into the z-smallest values. Therefore, an attacker cannot artificially increase the result. There is no deflation protection in this scheme, because the authors argue that an attacker will only try to produce fake events, such as a fake accident, and not try to hide events.

Using cryptographic security in a different way, Palomar et al. [131] propose to use proof-of-work mechanisms [58] to hinder false event dissemination. The mechanism relies on RSUs to periodically send so-called puzzles to vehicles driving by. Vehicles store the received puzzles and solve them whenever they want to disseminate an event warning. Then, the warning including the solved puzzle and a signature on both are disseminated. Solving the puzzles requires a moderate amount of computational power to keep attackers from flooding the network with lots of spurious warning messages. However, the scheme cannot completely prevent insider attackers from disseminating false messages and must, therefore, be combined with other mechanisms.

Data-centric event validity.

Rather than relying on cryptography, Kim et al. [100] present a data-centric mechanism to judge event validity. A total of five data-centric validity detectors are presented in the paper: geographic position of the event, alignment with local sensors, behavior of other vehicles, and validation by RSUs. In addition, cryptographic signatures are checked to filter outsider attackers. Like Petit et al. [137], the authors use the remaining distance to the claimed event in their calculation to amount for the time-criticalness of safety messages. Similar data-centric approaches for different kinds of safety-related mechanisms have been proposed by other authors [e. g., 29, 30, 90, 109].

Many promising schemes exist that use cryptography to achieve resilience of event detection against insider attackers. But event validation is characterized by a major simplification: the actual information element, i. e., the event, is assumed to be agreed on using a mechanism not covered or protected by the validation mechanism. For instance, an accident event happens at a certain spot of the road and can be observed by all surrounding vehicles simultaneously within a reasonable margin of error. Having agreed on the event, aggregation is only performed for the actual integrity protection. For instance, signatures are collected, sometimes aggregated, and disseminated together with the event.

5.4 RESILIENT AGGREGATION

The schemes discussed in Sections 5.2 and 5.3 either discuss security mechanisms for aggregation protocols in a different network setting or discuss integrity protection and validation for a much simpler form of aggregation. In the following, we discuss resilient aggregation mechanisms tailored to VANETs. In [122] and [23], a subset of the mechanisms discussed here has been surveyed. We base our discussion on these surveys but provide a broader overview of the field, as well as more extensive descriptions of the methods used.

Interactive attack detection.

Picconi et al. [141] proposed one of the first mechanisms to secure aggregation in VANETs. Their mechanism is based on probabilistic, interactive verifications. Whenever a car receives an aggregated record, it reads the claimed number of participants n. A random number $r \in \{1, \ldots, n\}$ is selected, and the sender of the aggregate is challenged to provide the r-th atomic observation, including a signature of the original observer as proof that the aggregation was performed correctly. Merkle trees [225] could be used to ensure that the sender does not provide other values with index $r' \neq r$, but the authors do not discuss this explicitly. Given a correctly signed atomic observation, the receiver checks whether it is plausible that the atomic observation was used in the aggregate. Typically, this means that the location must be within the aggregate area, amongst other checks. If an attacker takes into account some atomic observations but alters others, it is detected with probability f/n where n is the number of claimed atomic observations in the aggregate and f the number of fake observations.

To avoid the protocol's interactivity, the authors propose to use a tamper-proof device. The tamper-proof device acts as a proxy for the receiver inside the sender's car. To send an aggregate, the sending car forwards it to its own tamper-proof device. The device then performs the random challenge and broadcasts the results. Even if the software outside the tamper-proof device cannot provide a response to the challenge, the aggregate is sent as a proof of malicious behavior. It is therefore crucial that an attacker cannot prevent the tamper-proof device in its own controlled car from sending packets. Otherwise, the malicious behavior proofs can be kept from being sent. Given the numerous possibilities of jamming attacks, this guarantee is hard to achieve.

Ibrahim et al. propose a security mechanism tailored to their CASCADE protocol (see Section 3.5). Ibrahim and Weigle [84] and Ibrahim et al. [86] assume that trusted computing is employed to tamper-proof the aggregation algorithms, i. e., fusion, decision, and dissemination. Therefore, an attacker cannot modify information about larger areas. Putting the whole aggregation algorithm inside tamper-proof hardware is a possibility that Picconi et al. discussed as well. However, they argue that maintaining the whole aggregation logic inside tamper-proof devices is too costly.

Assuming the aggregation process to be tamper-proof, Ibrahim et al. only assume the on-board GPS devices to be unprotected. That enables an attacker to disseminate false location information. To detect these attacks, a combination of signal strength measurements and additional laser distance measurements is employed. This is an example for purely plausibility-based attack detection, and it only works in the vehicle's local vicinity, which explains why the authors employ trusted computing to secure dissemination of aggregated reports in wider areas. To exclude misbehaving vehicles from the network, quarantine messages are employed. These messages are received by a trusted component in attacker vehicles which then disables the attacker's radio devices.

Protecting fixed-segments aggregation.

The approaches we discussed so far use interactive detection, plausibility-based detection, or a combination thereof. However, both approaches rely on trusted hardware. In contrast, Raya et al. [145] propose a mechanism that employs cryptographic protection mechanisms without the need for trusted hardware. Their scheme assumes that the underlying aggregation mechanism works similar to `FIX`, which we discussed in Section 3.7.1. Having agreed on an average value per road segment, the goal is to protect that value against further modification. To achieve this protection, Raya et al. discuss three different signing mechanisms, which trade off computational overhead and communication overhead. Fundamentally, the scheme is limited by the requirement of fixed segments and non-hierarchical aggregation. In all variants of the signatures, unique identities are required to thwart Sybil attacks, reducing driver privacy.

For a detailed introduction of [13] see Chapter 7.

Molina-Gil et al. [123] enhance one of our schemes [13]. The basic idea is to use a number of original atomic observations including signatures, so-called witnesses, to prove the validity of aggregates. Witnesses should be selected to be equally distributed within the aggregate's geographic area. Also, witness information should be consistent with the aggregated information. For instance, witness velocity should be similar to the aggregate's average velocity. Molina-Gil et al.'s extensions introduce probabilistic verification to cope with the processing load due to signature verification. For each message received, only a random subset of the presented signatures is checked. Further, the authors propose to adjust the granularity of included atomic observations according to the type of road. For instance, less observations should be added on highways.

In a follow-up paper [124], Molina-Gil et al. introduce a group creation phase to their scheme. When a vehicle detects an event, it creates a cluster of vehicles and elects a cluster head. The cluster head then collects information from its cluster members and disseminates the aggregated result back to the clus-

ter members. Then, all cluster members check the received aggregate for consistency with their local information and sign it. The cluster head attaches all signatures as witnesses to the aggregate and further disseminates it. Receiving vehicles only probabilistically check the received signatures, as in their previous scheme. Despite the agreement on a common message, the authors do not propose to use more efficient signature schemes such as multi-signatures [31] or aggregate signatures [34, 35]. Therefore, their scheme uses significant overhead to secure aggregated information.

Protection of FM sketches.

All mechanisms discussed so far can be applied to arbitrary fusion functions. However, many aggregation mechanisms specifically employ various kinds of sketches for data fusion [e. g. 114]. These data structures offer desirable properties, like duplicate insensitivity (cf. Section 3.5). Consequently, several newer secure aggregation schemes explore the possibilities to secure sketches against manipulation, continuing a trend we have discussed for event validation in Section 5.3.

Han et al. [78] secure aggregation schemes where an average value, e. g., a speed average, is calculated for fixed road segments. They base their protection on secure FM sketches. Moreover, the authors assume that vehicles communicate their reports exclusively to roadside units, which forward them to a centralized TMC. Thus, the vehicles are only assumed to share a key with the TMC. Only the TMC – and not vehicles inside the network – can verify the proofs on the presented aggregates. Using these assumptions, the authors propose to employ symmetric cryptography, i. e., message authentication codes (MACs), as signatures. Relying on such centralized infrastructure, however, hinders the bandwidth saving of in-network aggregation, which benefits especially from distributed dissemination.

For inflation protection, the same approach as in Garofalakis et al. [72] is used: whenever a vehicle sets one of the FM sketch bits to 1, it attaches a signature on the event id, event segment, time, and vehicle ID. However, some of the signatures are merged to save additional space. Namely, signatures on the initial uninterrupted sequence of 1 bits in the FM sketch are combined. Only signatures on 1 bits that are not part of the initial sequence are kept separate, because the position of these bits cannot be predicted. To achieve deflation protection, the vehicle that calculates the aggregate initializes a hash chain using it's signature on the aggregate segment ID and time. Then, a one way function is applied i times where i is the length of the initial 1 bit sequence, forming a hash chain. If the aggregate is merged with other aggregates, and the 1 bit sequence gets longer, any vehicle can apply the one way function again. However, an attacker cannot invert the one way function to deflate the sketch estimate. The TMC can recalculate the initial signature on the event and time and verify the hash chain, because it shares a key with each car. The size of the hash chain is constant, as opposed to the combined size of all signatures on the complement FM sketch in Garofalakis et al.'s proposal.

Especially schemes that use FM sketches exhibit a considerable overhead due to the amount of added signatures. Wang and Tague [165] propose an al-

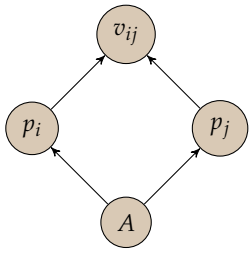


Figure 5.3: Basic consistency checking structure in [165].

ternative to cryptographic security. Their approach uses statistical measures combined with a directed acyclic dissemination graph that includes redundant dissemination paths. Figure 5.3 shows the verification methodology. An aggregator A sends a value towards a verifier v_{ij} via two independent paths. Both intermediate nodes, p_i and p_j, merge A's value with their individual atomic observation. They then forward the merged value and their own value. v_{ij} then reconstructs A's original value using the values received. In case p_i or p_j modified the value independently, their attack will be detected. To further prevent collusion of p_i and p_j, the authors use a modified version of *Timed Efficient Stream Loss-tolerant Authentication* (TESLA) [136]. TESLA is a time-delayed source authentication protocol based on symmetric cryptography, which is used here to authenticate A's original value. Drawbacks of the scheme are time delays introduced by TESLA, as well as the assumption that aggregation is initiated by a designated querier vehicle rather than using a push paradigm.

5.5 SUMMARY

We have discussed related work from the WSN domain, as well as the VANET domain, including mechanisms for event validation. Mechanisms from the WSN domain are not directly applicable to in-network aggregation in VANETs due to the different network characteristics. The most interesting result in the area of secure WSN aggregation is the lower bound for single-round insider-secure aggregation protocols presented by Bhaskar et al. [28]. Their results indicate that a perfect solution for resilient in-network aggregation cannot exist, and that trade-offs and adaptive, probabilistic mechanisms are required.

Even though solutions for event validation in VANETs are promising, they need to be significantly adapted to deal with the collaborative detection of real-world phenomena used in in-network aggregation. In fact, VANET in-network aggregation schemes that use cryptographic security mechanisms try to recreate the setting of event validation schemes. A common approach is to divide the road into fixed segments [e. g., 78, 145], in which an agreement on aggregate values before signing is assumed. Other mechanisms employ cluster structures to enable interactive agreement on a common value [e. g., 124]. All these mechanisms restrict the flexibility of aggregation and potentially reduce data quality, which are two main requirements for in-network aggregation, as discussed in

Section 3.4. The aim of our work is to lift these restrictions and provide security mechanisms that are applicable to more flexible aggregation approaches.

Finally, related work shows that more flexibility in the underlying aggregation scheme correlates with higher security overhead [e. g., 123]. Therefore, some approaches for event validation [e. g., 100] incorporate data-centric approaches, which do not require extra cryptographic overhead. Some in-network aggregation mechanisms employ data consistency checks, too [e. g., 123, 124]. But in their case, checks are restricted to comparing witness information to aggregated values. Therefore, a second aim of our work is to explore more elaborate models and approaches for data-consistency checking.

METHODOLOGY

6

6.1 OVERVIEW

Based on the analysis of related work, we focus our research on security mechanisms that support flexible aggregation schemes, support adaptivity, and use data-centric mechanisms to avoid cryptographic overhead where possible. In this chapter, we describe our research methodology, which entails two aspects. First, Section 6.2 presents a structured research approach that guides our mechanism design. We use the promising security approaches identified in Chapter 4 as a basis to ensure that our security mechanisms address a wide range of security paradigms. This structured approach is the basis for a detailed evaluation of all presented security mechanisms regarding their applicability in different scenarios.

Our evaluation method, which we discuss in Section 6.3, is predominantly based on network simulations and complemented by a formal overhead analysis. The goal is to apply our security mechanisms in realistic traffic scenarios to determine whether they can detect the representative attacker types we identified in Chapter 4. The use of network simulations allows us to use a realistic algorithmic implementation of our mechanisms, which could also be deployed in real testbed vehicles, and test it with large numbers of vehicles to assess the mechanisms' scalability. The evaluation focuses on two aspects: resilience against attackers and scalability, which is determined by the mechanisms' bandwidth overhead. We use our definition for in-network aggregation resilience from Chapter 4 to quantify the mechanisms' resilience.

Section 6.4 complements the research approach and evaluation method with an overview of the specific mechanisms we will present in more detail in the following chapters.

6.2 RESEARCH APPROACH

In Section 4.6, we introduced four types of security paradigms:

1. cryptographic mechanisms,
2. interactive mechanisms,
3. data consistency mechanisms, and
4. trusted-hardware-based mechanisms.

As discussed in Section 4.6.4, we argue that mechanisms based on trusted hardware are too costly and unlikely to be supported by upcoming VANET deployments. Therefore, we do not pursue this category further. The other three categories, however, each have their individual merits and shortcomings. To address a wide range of security requirements and application scenarios, as well

as to explore a number of trade-offs between security and bandwidth overhead, we therefore propose multiple security mechanisms that leverage these three basic security paradigms. In particular, we present two mechanisms that rely primarily on cryptographic protection and complement it with data consistency checks in Chapter 7 and Chapter 8. In Chapter 9, we present a clustering approach that contains an integrated aggregation protocol and is the basis for an interactive security mechanism we propose. Finally, we explore a mainly data-centric approach in Chapter 10.

Mechanism presentation structure.

We assess the security properties of each mechanism, and we evaluate its resilience, information quality conservation, and bandwidth overhead. We acknowledge that no single mechanism presented in this thesis is likely to provide a complete solution for resilient in-network aggregation in all situations. Rather, each mechanism is suitable for different situations and provides different trade-offs. Based on the separate mechanisms, we show how their combination and adaptivity in a generic architecture can improve detection results and provide a comprehensive solution for resilient in-network aggregation in Chapter 11. Before presenting our approaches in more detail in subsequent chapters, we outline our general methodology for evaluating their effectiveness.

6.3 EVALUATION METHOD

The major goal of resilient in-network aggregation mechanisms, which is characterized in our central research question (see Section 1.1), is to protect information integrity against insider attackers while maintaining the bandwidth savings introduced by aggregation. Our evaluation approach reflects these goals. We evaluate our mechanisms' resilience against attackers and compare it with the overhead that is required to achieve this resilience.

We choose highways as our main evaluation scenario, which we explain further in Section 6.3.1. To implement our evaluation scenarios, we use network simulations. Our network simulator entails simulation of a realistic road network with vehicle movement, and it simulates a realistic physical layer and MAC layer for each vehicle. Each vehicle runs a proof of concept implementation of our mechanisms. The network simulator runs our implementation on each vehicle and simulates realistic message transmissions using the provided lower network layers. This implementation enables to study our mechanisms' behavior in complex application scenarios and with higher numbers of vehicles compared to using real deployments for tests. We discuss our simulation setting in Section 6.3.2.

To quantify the abstract resilience goal, we use the resilience definition that we developed in Section 4.3.2. That is, we quantify the extent to which the attacker-influenced information makes other vehicles' world models deviate from the correct situation. The quantitative resilience metric is applied to the simulation results we collect using the network simulation tool. To quantify bandwidth overhead, we use a practical estimation of common traffic information system application requirements. The bandwidth overhead quantification

is applied to network simulations, as well. We explain these two main evaluation metrics in Sections 6.3.3 and 6.3.3, respectively. Comparing the mechanisms' quantitative resilience and overhead allows us to evaluate to what extent mechanisms are applicable in different scenarios, such as with varying traffic density or with varying numbers of attackers. Moreover, quantitative metrics allow to reason about trade-offs between resilience and bandwidth. For instance, a mechanism with slightly lower resilience but very little bandwidth overhead may be favorable for certain applications over more resilient mechanisms with high bandwidth overhead.

Besides quantifying resilience and bandwidth overhead, we use our evaluation to derive adaptation parameters. Each mechanism we propose offers certain parameters that lead to higher resilience at the cost of higher bandwidth overhead. We study the effect of these parameters in our evaluation to facilitate suitable mechanism combinations and propose adaptation strategies in Chapter 11.

While our evaluation method allows to judge the resilience and effectiveness of our mechanisms, we acknowledge that it has limitations compared to testbed deployments. We discuss these limitations in Section 6.3.4.

6.3.1 *Evaluation scenarios*

We choose highways as main evaluation scenario for our mechanisms for two reasons. First, we argue that vehicles on a highway can benefit most from an aggregation mechanism that is based predominantly on vehicle-to-vehicle communication. Second, using highways ensures comparability of our results to existing works on both aggregation mechanisms that focus on scalability rather than security and resilient aggregation mechanisms.

It is unlikely that highways will ever be fully equipped with supporting infrastructure, such as roadside units or full cellular network coverage. Therefore, highways can benefit most from independent protocols that solely rely on vehicle-to-vehicle communication. Moreover, highways require dissemination of information within large geographic areas – in the order of dozens of kilometers –, which makes aggregation particularly applicable. In city scenarios, it is more likely that additional infrastructure will be deployed, which may require different, hybrid aggregation approaches. At this point in time, it is hard to estimate such infrastructure deployments in city scenarios, but it is feasible to estimate the situation in highway deployments.

These characteristics also reflect in the state of the art of unsecured aggregation protocols, as well as secure aggregation protocols: the majority of existing works focuses on providing traffic information in highway scenarios [2]. Therefore, using highways for simulation of our protocols enables comparability and allows to use existing insecure aggregation protocols to build our security mechanisms upon where applicable.

The simulated highway stretches vary in length between 1500 m and 10 000 m, depending on the characteristics of the individual security mechanisms. For

Table 6.1: Overview of simulation parameters.

Parameter	Value
Discreet event simulator	JiST
Network simulator	SWANS
Path loss	Two ray ground
Fading	Rayleigh
Transmit power	10.9 dB
Noise model	Additive
MAC	IEEE 802.11p
Beaconing interval	1000 ms
Vehicle placement	Random
Mobility	Car-following
Scenario type	Highway
Scenario length	1500–10 000 m
Number of vehicles	30–1000
Random seeds	10

instance, we simulate longer stretches of highway for mechanisms that focus on particular on scalability to large numbers of vehicles. Moreover we simulate different traffic situations for each security mechanism. Traffic situations represent low, medium, and high vehicle density, and they correlate with free-flowing traffic, slow-moving traffic, and congested traffic, respectively.

Investigating these different traffic situations shows how mechanisms are influenced by different network characteristics, such as traffic homogeneity, relative velocity, and number of vehicles within single hop broadcast range. The simulation results for these scenarios are used to evaluate each mechanism separately, as well as to derive adaptation and configuration parameters for the potential combination of several security mechanisms.

6.3.2 *Evaluation implementation*

Example field tests are the DRIVE-C2X project (www.drive-c2x.eu) and the Score@F project (team.inria.fr/scoref).

Although a number of field operational tests are underway as of 2015, network simulations are the predominant evaluation tool for VANET research in general and VANET security mechanisms in particular [e. g., 99, 230]. Network simulations allow to evaluate the integration of network protocol behavior with vehicle mobility at a large scale. Especially for traffic efficiency applications, large numbers of vehicles are needed to assess the effects of aggregation. Moreover, the security approaches discussed in Section 4.6 typically rely on cooperation of larger groups of cars to detect attacks. Therefore, we choose network simulations to demonstrate resilience, as well as scalability of our proposed mechanisms.

To implement our simulation scenarios, we use the JiST/SWANS network simulator [185]. JiST/SWANS is on of the three most-used network simulation

tools in the VANET research domain [92], which ensures that our simulation results are comparable to a large number of existing works.

The JiST/SWANS simulator is comprised of two parts. JiST introduces the simulation time concept to Java by intercepting method calls using Java's reflection API. SWANS models the network communication stack. Table 6.1 summarizes the main simulation parameters. JiST/SWANS' physical layer implementation is a re-implementation of the physical layer model in GloMoSim [180], which is another mobile network simulator. Both path loss and fading models are provided to determine reception of packets. We use the two-ray ground radio propagation model [208, Ch. 2] to account for path loss. In addition, Rayleigh fading [159] is applied to model fading effects. Further, an additive noise model is used to calculate model packet collisions. Takai et al. [163] discusses the different implementations of physical layer models in GloMoSim and other network simulators and their effects on modeling accuracy.

For the MAC layer, we use an implementation of IEEE 802.11p [210]. To simulate traffic on a highway, our predominant evaluation scenario, we use a self-developed microscopic car-following model. Initially, each vehicle starts with a speed that is sampled from a gaussian distribution around a pre-configured average speed. Each vehicle then aims to reach a target speed that relates to the highway's speed limit. Vehicles brake and overtake where necessary to avoid accidents. Braking and acceleration are guided by maximum values for the velocity difference per time unit.

The simulated physical layer, MAC layer, and vehicle movement are the basis upon which we build our mechanism implementations. For each security mechanism that we present in Chapters 7 to 11, we create a complete proof-of-concept implementation of all relevant algorithms and protocols. That is, we create working implementations for each mechanism's Decision, Fusion, and Dissemination methods. The network simulator executes these implementations on each simulated vehicle, and the provided physical layer and MAC layer implementations simulate realistic transmission of messages that takes into account the physical environment and vehicle movement.

In addition to the aggregation functionality, we implement each proposed security mechanism. That is, all vehicles attach the required signatures, as well as all other required meta-data. Also, each vehicle performs the proposed checks and verifications and processes or discards information accordingly. Note that our simulations focus on realistic representation of the security overhead rather than correct simulation of computational complexity. That is, we assume that sufficient processing power is available in future VANET hardware.

In addition to the regular mechanism implementation, we represent our representative attacker models in the simulations. That is, a number of vehicles in each simulation do not adhere to the regular algorithms but create, process, and disseminate messages that aim to fulfill the attacker goals. The specific implementation of the attacker is tailored to each security mechanism. Therefore, we describe details of the attacker implementation separately for each security mechanism evaluation. Moreover, we implement each security mechanism

and attacker such that the underlying aggregation mechanism can operate separately. Using this modular approach, we can assess the security mechanisms' resilience and overhead relative to the resilience and overhead of the underlying, unsecured aggregation mechanism.

We account for the dynamicity of VANETs by repeating our simulation experiments in multiple (typically 10) runs with different random vehicle placements and behavior to ensure statistical reliability of results. For each simulation, we show the arithmetic mean of all simulation runs, as well as the standard deviation. During the simulation, we collect all parameters that are required to evaluate resilience and bandwidth.

6.3.3 *Evaluation metrics*

Our main evaluation metrics are the resilience against attacks and the overhead of our proposed security mechanisms. In Chapters 3 and 4, we discussed qualitative requirements for resilience and overhead. To compare our security mechanisms, we use our qualitative requirements as basis for quantitative metrics. Namely, we introduce a quantitative resilience metric and a quantitative overhead metric. Both metrics can be applied to the parameters we collect during the execution of our network simulations.

Quantification of resilience

The evaluation approach we adopt is related to the evaluation methodology of intrusion detection systems (IDSs) [e. g., 112]. We use the percentage of attacks detected together with false negatives (i. e., undetected attacks) and false positives (i. e., false alarms) for assessing the quality of our approaches. We compare those results with the ground truth; that is, normal simulated system operation in absence of attacks. More specifically, our simulative evaluation uses the following scheme:

1. we simulate the underlying aggregation scheme without attackers and without security countermeasures to acquire a baseline;

2. we simulate the aggregation scheme with attackers but *without* countermeasures to gauge the attack influence and validate the attacker implementation; and

3. we simulate the scheme with attackers *and* countermeasures to assess how resilient the proposed schemes are against attackers, but also how much they impact performance.

The comparison of these three settings establishes the relative validity of our simulations. By executing the underlying aggregation mechanisms without security or attackers enabled, we acquire a baseline or control measurement. The representation of the traffic situation in the baseline should closely resemble the actual traffic situation as it would be observed by collecting all un-aggregated

information from all vehicles. Thereby, evaluating this setting ensures that the underlying aggregation mechanism is implemented correctly. Next, we add attackers but leave the security mechanism disabled. In this setting, the derivation from the actual traffic situation should be large. This comparison established the correct algorithmic implementation of the attacker. Finally, we add our security mechanism. The observed simulation results then assess whether the security mechanism successfully represents the real traffic situation as good as the setting without attackers. By simulating all three settings, we ensure that our security mechanisms improve the situation representation in face of attackers that would otherwise successfully influence the representation.

To determine attack detection, false positives, and false negatives, we need to quantify resilience. In Section 4.3, we defined a measure for resilience of in-network aggregation. We use this definition to define an attacker's success based on the achieved deviation of the attacker-generated information from the real world situation. As we argued in Sections 4.4.2 and 4.4.3, the primary attacker goal is to create inexistent situations that deviate from the real world situation. For instance, an attacker may want to create aggregates pertaining to a traffic jam where there is none in the real world. Therefore, the attackers' success can be quantified by the average amount of deviation they create. We base our deviation measurement on the converged world model of each vehicle at the end of our simulations. As the goal of in-network aggregation is to represent macroscopic traffic situations rather than microscopic behavior of vehicles, we do not to take into account changes in the world model over time. Rather, we consider attackers to be successful if they are able to introduce a deviation from the real world situation that is persistent once the world model has converged.

Resilience criteria.

An aggregation mechanism is resilient if attackers can create no or only negligible deviation. In contrast to traditional IDSs, attack success, false positives, and false negatives are not binary but gradual values. For instance, an attack may lead to a deviation of 20 km/h between the perceived traffic velocity and the real situation.

Figure 6.1 shows an example evaluation plot. The three lines represent the three simulation scenarios outlined above. Each line shows the average perception of the situation. It is calculated based on the honest vehicles' world models and averaged over multiple simulation runs to ensure stability of results. Ideally, the attack with enabled countermeasures does not deviate significantly from the baseline aggregated view, as shown in the figure. In addition to the average deviation, the standard deviations help to determine the security mechanisms' performance. In case the standard deviation is high in the aggregated view without attack, that may be a sign for inconsistent real world situation representation by the aggregation mechanism. In case the standard deviation is high only in the scenarios with attackers, it indicates that attackers are successful in some cases but are not consistently able to alter the perceived situation.

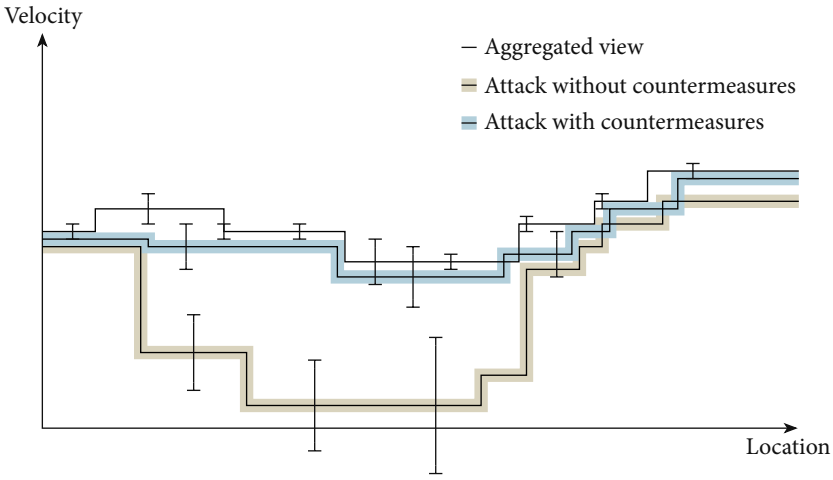


Figure 6.1: Example attacker influence evaluation plot.

Quantification of overhead

Because in-network aggregation mechanisms combine and merge information from multiple vehicles, integrity protection may be complex and may require high bandwidth overhead. In our proposals we consider all information that is added to achieve protocol resilience as security overhead. Such additional information may be cryptographic signatures, key material, certificates, or additional atomic observations or aggregates used for data consistency checks.

Scheuermann et al. [152] provide a theoretical bound for the bandwidth consumption of in-network aggregation mechanisms that are expected to scale to infinitely large regions. However, Scheuermann et al. also note that infinite scalability may not be required for practical applications. For our evaluation, we therefore adopt an application requirements driven approach for bandwidth overhead quantification.

See Section 3.4.1 for a discussion of Scheuermann et al.'s result.

Essentially, the goal of in-network aggregation is to make information about a *sufficiently large* geographic area known with *sufficiently low* overhead. Next, we develop a quantification for this intuitive characterization.

To determine the required geographic area size, we consider a traffic situation for a highway, which is one of the predominant use cases for in-network aggregation, as discussed in Section 3.3. The minimum goal of such a traffic information system should be to provide drivers with the opportunity to choose alternative routes to avoid traffic jams. To react in time, drivers should therefore *at least* be aware of the traffic situation between two consecutive highway exits. Along the German highway A7, Europe's longest national highway [186], the minimum distance between two exits is 0.4 km and the maximum distance is 19.1 km ($\mu = 7.18$ km, $\sigma = 4.03$ km) [231]. Therefore, a minimum area coverage of 7–20 km should be achieved to satisfy application requirements.

We consider the maximum size of a single network packet – i. e., the so-called MTU – as maximum tolerable overhead to achieve this geographic coverage. If information about the required geographic region cannot be expressed in a single message, more complex scheduling strategies would be required that fragment information into multiple packets and transmit them at different points in time. As such scheduling would introduce additional protocol complexity, it should be avoided. The IEEE 802.11p standard [210] foresees an MTU of 1500 bytes for VANETs. Therefore, we use the achievable geographic area coverage using at most 1500 bytes as a guideline to evaluate bandwidth overhead.

6.3.4 *Limitations*

Using network simulations allows to assess the resilience and overhead of our mechanisms in realistic scenarios. Moreover, network simulations enable to simulate large numbers of vehicles that would not be feasible to use during field operational tests. We ensure the validity of our results by using JiST/SWANS as network simulation tool, which is widely used in the VANET research community. And we ensure relative validity of our results by comparing different simulation settings with and without attackers and with and without security measures enabled, as explained in Section 6.3.3.

However, we are aware that network simulations introduce abstractions that may lead to differences between our simulation results and evaluation of our mechanisms using actual vehicles in testbeds or even during live system operation. For instance, communication may be influenced by physical layer influence that is not fully represented in the simulation environment, such as trees, buildings, or bridges, which may obstruct signal transmission. However, such obstacles would influence the underlying aggregation mechanisms in the same way they influence our proposed security mechanisms. Therefore, absolute results for resilience or bandwidth use may differ, but it is likely that the relative performance of our security mechanisms would remain stable.

Moreover, we assess a wide range of different highway traffic situations, including free-flowing traffic and congested traffic, but we focus our simulations on highways. As argued in Section 6.3.1, highways will likely benefit from aggregation. While focusing on highways ensures comparability with existing work, simulations in city scenarios may provide different results for resilience or bandwidth usage. We argue that such differences can be adapted to using our mechanism combination framework, which we present in Chapter 11. Therefore, our mechanisms can conceptually be adapted to city scenarios, but we acknowledge that our simulation results cannot be used to derive absolute statements about the performance of each individual mechanism in cities.

We conclude that our evaluation method allows to derive relative statements about the resilience and overhead of our mechanisms in highway settings, which are a prime deployment target for in-network aggregation mechanisms. But our evaluation would need to be complemented with field operational tests using large numbers of vehicles in different scenarios before the security mechanisms

can be applied in real deployments. Future deployments of basic VANET applications are likely to provide the basis for such tests in the future.

6.4 MECHANISMS OVERVIEW

Our mechanism proposals are based on the categorization of security approaches (Section 4.6) and are grounded in our discussion of related work on secure in-network aggregation in Chapter 5. In Chapters 7 to 10, we propose four security mechanisms, and in Chapter 11, we propose a combination and adaptation framework that is derived from the evaluation of our proposed mechanisms.

The design of resilient aggregation mechanisms often requires the tight integration of application knowledge and protocol design. For instance, identifying a stretch of homogeneous traffic is the goal of an aggregation mechanism for traffic information systems. At the same time, the stretch of homogeneous traffic allows to use interaction between vehicles to reach mutual agreement on aggregated information. Whenever tight integration of the use case and the security mechanism allows to considerably reduce bandwidth use or improve attack detection, we focus our discussions on the traffic information system use case, which we identified as the dominant application scenario for in-network aggregation in Section 3.3.

The use of cryptography is a widespread approach to secure VANET protocols against insider and outsider attackers. When used to protect against insider attackers, cryptographically signed observations often serve as witnesses for events. The more witnesses from different vehicles validate an event description, the more likely it is that the event description is correct. Data consistency checks are used in combination with signatures to implement validity checks, such as homogeneity of velocities reported by witnesses.

In aggregation protocols, events often span large geographic areas, and many witnesses may potentially attest to the correctness of event descriptions. As a result, cryptographic resilience mechanisms face two main challenges. First, only a subset of all potential witnesses should be chosen to attest to event descriptions. The subset should be as small as possible while maintaining resilience against insider attackers. Second, witnesses originally attest to atomic observations, which are later changed in the aggregation process. Therefore, mechanisms either need to implement ways to agree on a common value that witnesses can sign, or they need to implement data consistency mechanisms to correlate signed atomic observations with aggregated information.

Related work often implements restrictions on aggregation flexibility to cope with those requirements. Raya et al. [144] assume an underlying aggregation mechanism similar to `FIX`. Then, vehicles within fixed segments can agree on a common value before signing it. Likewise, Garofalakis et al. [72], Han et al. [78], and Hsiao et al. [82] use fixed aggregation structures to agree on values and select witnesses. We propose two mechanisms for mainly cryptography-based resilience, which assume an underlying aggregation mechanism similar to `DYN` and thereby allow for more flexible aggregation decisions.

SELECTIVE ATTESTATION Our first mechanism (Chapter 7) chooses a subset of all atomic observations $o_i \in P^{\bowtie}(A)$ that led to an aggregate A to serve as witnesses. Selection depends on the geographic area covered by an aggregate and aims for uniform distribution of witnesses throughout the aggregate area. Instead of explicit agreement on aggregated values before signing, we use data-consistency checks that correlate atomic observations that serve as witnesses with the aggregates. Thus, our scheme can be applied to dynamic aggregation. While we exemplify the mechanism for traffic information systems, the selective attestation may also be applied to parking spot counting, where it could serve as indication for minimum parking spot availability.

MULTI-SIGNATURES Our second mechanism (Chapter 8) is inspired by the existing approaches that use FM sketches to secure in-network aggregation [e. g., 72, 78]. Our main goal was to maintain the overhead reduction and duplicate elimination properties of FM sketches while allowing for more flexible aggregation decisions than existing work. The result is a two-round protocol. In the first phase, secured FM sketches are used to protect a dynamic aggregation phase, which also serves as agreement protocol for aggregated values. In the second phase, multi-signatures are used for witnesses on the previously-agreed aggregates to conserve bandwidth. Like selective attestation, the multi-signatures-based approach focuses on traffic information systems. However, Lochert et al. [114] have demonstrated the applicability of FM sketches to parking spot counting. By combining our sketch protection approach with their communication protocol, the security mechanism could be applied to parking systems.

Both cryptography-based mechanisms allow for dynamic aggregation decisions. As a downside, the required security overhead of both mechanisms is linear in the geographic area covered by the aggregates. Therefore, we explore interactive security in our third proposed mechanism. The underlying idea is that existing work [e. g., 144] uses clustering for agreement on aggregated values, but limits aggregation to fixed segments. We introduce a novel clustering protocol for VANETs, which allows to build stable groups of vehicles that are determined by aggregation decisions such as homogeneous velocity. Thereby, the clustered vehicles represent a dynamic, flexible segmentation of the roads, and at the same time, they are stable enough to implement interactive security protocols on top of those clusters.

CLUSTERING Our cluster-based mechanism (Chapter 9) leverages velocity-based clustering. We regard clusters as trustworthy units given that they have at least a certain number of cluster member vehicles. Within clusters, vehicles interact to securely agree on an aggregate that represents the traffic situation within the cluster. Between clusters, we use HyperLogLog sketches [69] for bandwidth efficient proofs of the number of cluster members. Hierarchical aggregation benefits from our assumption that clusters are trustworthy units, because proofs are only commu-

Table 6.2: Overview of mechanisms

Mechanism	Paradigm	Scenario size	Attackers
Selective attestation (Ch. 7)	Cryptography	1500 m	CONCEAL (0, 33 %)
Multi-signatures (Ch. 8)	Cryptography	5000 m	FAKE (0–10 %)
Clustering (Ch. 9)	Interactivity	5000 m	FAKE (0–100 %)
Redundancy (Ch. 10)	Consistency	3500 m	CONCEAL (1 %)
Combined (Ch. 11)	Combination	5000 m	FAKE (5–20 %)

nicated between directly adjacent clusters. While the protocol benefits from the combination of cluster formation and traffic situation detection, the same approach could be applied to agree on the sensed number of available parking spots in the region covered by a cluster.

The cluster-based resilience mechanism is particularly useful for situations with high traffic density where clusters are large. As an orthogonal approach to securing in-network aggregation, we consider redundancy in our fourth proposal. The assumption is that an attacker likely cannot influence all redundant information flows within the network, and therefore, data consistency checks can be used to reason about correctness of information. Such data consistency checks have been applied in other domains [38, 39]. But in-network aggregation complicates redundancy checks, because information is merged during dissemination, which complicates the definition of redundant information.

REDUNDANCY-BASED ANALYSIS Our redundancy-based security mechanism (Chapter 10) introduces path list filtering based on secured path lists to detect node-disjoint information flow paths, which represent redundant information flows in the in-network aggregation setting. Based on this filtering concept, we implement a statistical analysis that determines correct information and filters attacker-generated values. As the mechanism is based mostly on information redundancy, it is applicable to all kinds of transmitted information, including traffic information and parking spot availability.

The redundancy-based mechanism is especially applicable to scenarios with high information redundancy. Even in other situations, where it may not be able to determine which information is correct, it can still serve as a mechanism to detect presence of attacks.

Table 6.2 summarizes our proposed security mechanisms and their evaluation scenarios. Because our mechanism proposals address different scenarios and their overhead and resilience depends on context, we further propose a framework to combine and adapt mechanisms based on such context.

COMBINATION AND ADAPTIVITY (Chapter 11) Our framework combines results from all our mechanisms and implements two orthogonal adaptation strategies. First, resilience mechanisms are enabled or disabled

based on traffic context. This adaptation allows to leverage the benefits of cluster-based security in high traffic density scenarios while maintaining the flexibility of our cryptography-based mechanisms in other scenarios. Second, mechanisms are adapted to use varying amounts of bandwidth depending on current overall attack likelihood. This adaptation allows to use our redundancy-based analysis, together with other enabled mechanisms, as a baseline to detect possible attacks and only use larger amounts of bandwidth for security overhead when attacks are likely to happen.

With our four proposed security mechanisms, we contribute significant improvements over related work, focusing on dynamic and flexible aggregation. We propose mechanisms with different resilience strategies, which are, in combination, applicable to a wide range of different attack scenarios and traffic situations. To implement the combination of mechanisms, we propose a framework that allows adaptive combination of mechanisms, which leads to lower bandwidth overhead while increasing resilience against insider attackers.

6.5 SUMMARY

To ensure resilient aggregation in varying contexts, our methodology is based on our categorization of security mechanisms. We identify three promising types of security mechanisms, which we term cryptographic, interactive, and consistency-based, respectively. Our evaluation is based on our main research question and instantiated by a quantification of our resilience definition, as well as practical application requirements.

We have outlined how our specific mechanism proposals relate to existing work, and how they leverage different security paradigms. In the following chapters, we show the design and evaluation of our mechanism in detail, followed by our proposal for a generic combination and adaptation framework.

III

RESILIENT AGGREGATION MECHANISMS

7

7.1 OVERVIEW

Parts of this chapter (Sections 7.1–7.4) are a revised and extended version of an algorithm published in [13], which is based on the author's Diplom thesis [187].

The first security mechanism we present is based on cryptographic message protection in combination with data consistency checks. From a message integrity perspective, each aggregating vehicle would attach all information items used during the aggregation process to the created aggregate. Then, receivers could verify that the aggregation process was performed correctly. The only remaining attack is to forge atomic observations, which can be detected assuming an honest majority of vehicles. But as argued in Section 4.6.1, this approach is infeasible from a scalability perspective. Therefore, we construct a security mechanism that is still based on the idea of attaching information items used during aggregation but only attaches a subset of these items to help scalability. Using specific selection criteria, we can achieve a trade-off between resilience and scalability.

To ensure flexible aggregation, this security mechanism can be directly applied to the DYN scheme presented in Section 3.7.2. We use a traffic information system (see Section 3.3) that disseminates average speeds on different parts of the road as an example use case to make the scheme description more specific. Moreover, we exemplify our scheme for use on a single one-dimensional road. Thus, the format of an atomic observation is

$$o := ((p, t), v), \tag{7.1}$$

where p is the location, t the timestamp, and v the velocity; the format for aggregates is

$$A := (([a, b], [s, e]), v, \varsigma, c)\,; \tag{7.2}$$

Recall that $[a, b]$ is a geographic location interval, $[s, e]$ is the time interval, and ς denotes the standard deviation of averaged velocities.

and fusion is performed as shown in Algorithm 5. That is, locations and time intervals are merged such that the new intervals span the combined region and time, and velocities are averaged by calculating the arithmetic mean. Using more dimensions for the geographical locator or road ids, the scheme can easily be extended to report average speeds on whole road networks. Likewise, the scheme could be applied to other applications, such as parking information systems and weather warning systems in addition to the traffic information system we discuss here.

The basic scheme, which we will describe first, helps scalability but features considerable security overhead. We therefore extend the original scheme by introducing a number of optimizations that bring the overhead closer to the information-theoretic bound of *who signed what* (see Section 4.6.1) in Section 7.6, and we discuss parameters to adapt the scheme's overhead to current context in Section 7.7.

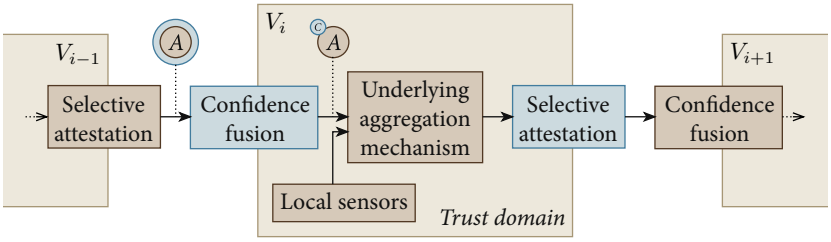


Figure 7.1: Overview of the system model showing the flow of aggregates and atomic reports through our system. Additions due to the security mechanism are highlighted.

The basic system model is shown in Figure 7.1. We assume that information originating from both local sensors, as well as other vehicles, enters a vehicle's local information base where it is stored, possibly further processed, and finally disseminated to other vehicles in the vicinity. Information originating from local sensors is considered to be correct whereas remote information needs to be further assessed to assign a certain confidence in its validity. For this assessment, we employ a combination of selective proofs using cryptographically signed atomic observations and probabilistic verification of the information contained in a given aggregate. This methodology results in a data-centric confidence [144] in contrast to a node-centric trust as it would result from a node reputation system.

Local trust domain.

As soon as a confidence value has been assigned to an aggregate, it is further processed locally using only this confidence value as basis for decisions by the underlying aggregation mechanism. We call this scope the local *trust domain* of a vehicle. Because a vehicle is assumed to trust its own confidence ratings, aggregates can be evaluated by examining their confidence values within the trust domain. However, as we do not employ a node reputation system, the trust domain does not extend beyond vehicle borders. Therefore, as soon as aggregates are selected for dissemination to nearby vehicles, the assigned confidence value loses its significance and will not be communicated. Receiving vehicles will again judge the aggregates according to the security mechanisms.

7.2 SELECTIVE ATTESTATION

A common technique to secure vehicular communication, especially beaconing applications, is to cryptographically sign each outgoing message, thereby giving a proof of the sending vehicle's authority to send messages, given that the accompanying public key is signed by a central, mutually trusted authority (see Section 4.2). If an adversary then sends out beacons with a high frequency to give his information a higher weight, other vehicles can easily detect the high frequency due to the attached signature and discard attacker messages. However, when using aggregation, atomic observations of several vehicles will be combined to aggregates. Even if all atomic observations were cryptographi-

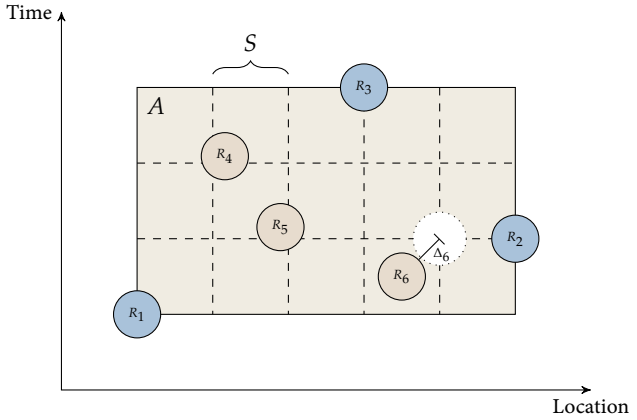


Figure 7.2: An example aggregate A with attestation meta-data attached. The attestation meta-data is comprised of three border atomic reports $\{R_1, \ldots, R_3\}$ and the additional reports $\{R_4, \ldots, R_6\}$ selected according to the granularity defined by the security parameter S.

cally signed initially, the signature cannot be verified after aggregation because the information that was signed has been altered, and one cannot reproduce the single speed reports given only the resulting speed average. Thus, deterministic verifiability is lost.

To achieve a certain verifiability of the aggregates nonetheless, we attach a certain amount of meta-information to aggregates, which serves as a witness for the correctness of the aggregation. This meta-information will be called *attestation meta-data*. A simple approach to select such witnesses would be to add all cryptographically signed atomic observations that served as input to the aggregation as attestation meta-data. In that case, any receiver could deterministically verify the aggregation by first checking all signatures of the atomic observations and then re-calculating the speed average and verifying that the result is the same as the one contained in the aggregate values. However, if all atomic reports are attached to aggregates as attestation meta-data, there will be no reduction of bandwidth usage compared to not using any aggregation at all.

Choosing attestation meta-data.

Therefore, we select a subset $W \subseteq P^{\bowtie}(A)$ of all atomic reports that served as input to an aggregate A to allow for a probabilistic verification. To stipulate the detection of malicious information with as few meta-data as possible, we employ a list of criteria to select the meta-data during the aggregation process.

As discussed in Section 7.1, the aggregates' locator is comprised of $[a, b] \subset \mathbb{R}^+$, which is a location interval relative to the road's origin, and $[s, e] \subset \mathbb{R}^+$, which is a time interval. Since all numbers are positive reals, the aggregate locator can be represented as a 2-dimensional rectangle $\square(a, s)(a, e)(b, e)(b, s)$ where location is the x-axis and time is the y-axis. Intuitively, our selection criteria aim to "cover" the aggregate locator area with selected atomic observations.

Algorithm 7: SelectMetaData(A)

INPUT: An aggregate $A = (([a,b],[s,e]),v,\varsigma,c)$ for which additional meta data should be selected.

RESULT: A set of atomic observations $W \subset P^{\bowtie}(A)$ that were selected.

$X \leftarrow \emptyset$ `/* grid points for location */`
$Y \leftarrow \emptyset$ `/* grid points for time */`
FOR $1 \leq i \leq \lfloor (b-a)/S \rfloor$ DO
  $X \leftarrow X \cup \{a + iS\}$
END
FOR $1 \leq j \leq \lfloor (e-s)/S \rfloor$ DO
  $Y \leftarrow Y \cup \{s + iS\}$
END
$W \leftarrow W_{\text{border}}$ `/* initialize selection with Equation (7.3) */`
FOREACH $x \in X$ DO
  $W \leftarrow W \cup \{\arg\min_{o=((p,t),v)\in P^{\bowtie}(A)} |p - x|\}$
END
FOREACH $y \in Y$ DO
  $W \leftarrow W \cup \{\arg\min_{o=((p,t),v)\in P^{\bowtie}(A)} |t - y|\}$
END
RETURN W

First, we can uniquely identify those atomic observations that led to an aggregate's current maximum and minimum in time and space (e. g., $\{R_1, \ldots, R_3\}$ for A in Figure 7.2). Namely,

$$W_{\text{border}} := \{o_i = ((p_i, t_i), v_i) \in P^{\bowtie}(A) : p_i = a \vee p_i = b \vee t_i = s \vee t_i = e\}. \tag{7.3}$$

Considering that aggregates will commonly represent an area in which vehicles share common values, e. g., common speed, all such atomic observations defining the borders of an aggregate's area will be added as attestation meta-data. Only if a vehicle is able to present valid signed information by distinct vehicles about all borders of an aggregate, it can be believed to be valid.

Attestation data with uniform distribution.

Especially if aggregates cover larger areas (as it is the case for A in Figure 7.2), adding values from the borders will only lead to a first indication that an aggregate is valid. An adversary can still select arbitrary atomic reports and craft an aggregate where the claimed values are only present at the borders and not throughout the area. This could, for example, lead to the CONCEAL attack discussed in Section 4.4.3. Therefore, additional meta-data is needed. The goal is to select each additional signed atomic report such that they are evenly distributed throughout the aggregate ($R_4, \ldots, R_6$ in Figure 7.2). The finer the granularity of the distribution, the higher the achieved security. To express this granularity, we introduce a system-wide security parameter S that defines the required granularity of the additional reports, marked by the dashed lines in Figure 7.2.

For now, we assume S to be a constant parameter. We will discuss in Section 7.7 how to adapt S to different contexts. Note that the matching to S may not be perfect. Reports are not available at arbitrary positions. They can differ from the ideal positions defined by S, e. g., R_6 differs by an amount of Δ_6, which is measured as the euclidean distance between the atomic observation's actual location and the ideal location according to the grid induced by S. Thus, checking the distribution of additional reports is not a binary process. Instead, the conformance to the ideal distribution can be characterized by considering all deviations Δ_i.

Algorithm 7 shows the selection process algorithmically for a case where a vehicle calculates a new aggregate A using a set of observations $\{o_1, \ldots, o_n\}$ that the vehicle received from its neighbors. That is,

$$A = o_1 \bowtie \ldots \bowtie o_n. \tag{7.4}$$

As shown, for each grid line in $x \in X$ that divides the aggregate's location and for each $y \in Y$ that divides its time, an atomic observation is selected that best matches the grid position in the aggregate. The selection can be applied to two-dimensional geographic regions by applying S to all dimensions. Including time, the aggregates would then have a three-dimensional locator, and S would define a three-dimensional grid.

Hierarchical aggregation process.

In practice, aggregation is an iterative, hierarchical process. Therefore, we proceed inductively to select additional reports during the aggregation process. Assume that two aggregates, A_1 and A_2, which already contain additional signed reports according to S, are selected to be combined. The area covered by A_1 and A_2 in time and space can either already overlap, or they can be disjunct. If they overlap, their contained additional reports suffice to achieve a good distribution throughout the aggregate. In the overlap region, only those additional reports are kept that are nearest to the optimal grid. To achieve the selection, Algorithm 7 can be executed using only the joint attestation meta-data of A_1 and A_2. If A_1 and A_2 do not overlap, their distance in time or space can be either smaller or larger than S. If the difference is smaller, all additional reports of A_1 and A_2 together will still approximate the ideal grid well. However, if the distance is larger than S, then there are regions for which no supporting reports can be found. The same considerations apply for the aggregation bootstrapping. When two signed atomic reports are first selected for aggregation and their distance is smaller than S, then the resulting aggregate will adhere to the criteria defined above. Otherwise, it will not. In all cases, we can find the necessary additional reports if aggregates or atomic reports selected for combination are not further apart than S.

Selecting the security parameter.

Thus, S needs to be selected such that it matches the underlying aggregation mechanism. If, for example, two reports will be aggregated when their distance is less than 500 m, then S needs to be at least 500 m. Smaller values for S will result in honest nodes not being able to adhere to the ideal grid for additional report selection. Larger values of S are possible, resulting in a lower probability of detecting attacks but also in a lower bandwidth overhead. Moreover, S needs

to relate to the underlying PKI. To increase confidence in aggregates, each selected atomic observation should be cryptographically signed with distinct public keys. However, the attacker may be able use multiple valid keys for signing at the same time, depending on the underlying pseudonym scheme. Then, S should be chosen such that the amount of attestation meta-data is larger than the number of distinct valid signatures a single attacker can create.

Considering the selection criteria for S and the attestation meta-data, an adversary can craft malicious aggregates in two ways. First, he can omit additional attestation meta-data that would otherwise expose the attack. This would lead to an uneven distribution of attestation meta-data. Receiving vehicles can detect this uneven distribution by comparing the received attestation meta-date with the grid induced by S. Second, an attacker can adhere to the distribution rules. Then, the values contained in the signed reports would make the forgery attempt obvious. In summary, we have defined two strategies for the selection of signed atomic reports serving as witnesses attesting the correctness of aggregates:

1. *Atomic observations from the aggregate borders* are always attached due to their exposed position. As the goal of aggregation is to combine reports from regions with similar characteristics, those values serve as cues for trust in aggregates.

2. Further, additional signed atomic observations that are *evenly distributed throughout the area of the aggregate* are selected to underline the correctness of the values. The bandwidth/security trade-off can be adjusted according to application requirements due to a configurable security parameter.

7.3 CONFIDENCE FUSION

The presented selective attestation mechanism results in cues leading to confidence in the correctness of an aggregate, namely:

BORDERS All borders of an aggregate are supported by a signed atomic report.

DISTRIBUTION Additional signed reports are evenly distributed throughout the aggregate, adhering to the security parameter S.

VALUE APPROXIMATION All values contained in the signed atomic reports presented as attestation meta-data support the claimed values of the aggregate.

In addition to the cues given by the selective attestation process, there can be a number of further cues that are defined by specific aggregation applications. One example is VELOCITY RANGE, i. e., the presented velocities are at most as high as the maximum speed of fast vehicles.

Due to the number of possible cues and their correlations with each other, their interpretation can be complex. Therefore, we employ a reasoning mechanism, namely fuzzy reasoning, to combine all collected cues into a single value that expresses the confidence in the correctness of an aggregate in percent. The methodology is similar to that applied to DYN's aggregation decisions in Section 3.7.2. Instead of evaluating multiple influences on aggregation decisions, however, we now evaluate multiple influences on confidence in aggregates. By using fuzzy reasoning [242], a number of influences can be combined using fuzzy rules without the necessity to define complex interrelations between the numerical values. In the following, we describe how fuzzy reasoning is applied in our mechanism. First, each of the cues needs to be expressed as a real number. For example, the above mentioned VALUE APPROXIMATION can be expressed by the root mean square error of all atomic reports' velocity values $v_1, \ldots, v_n$ and the claimed average value v of the aggregate:

$$\left(\frac{1}{n} \sum_i (v_i - v)^2 \right)^{1/2} . \tag{7.5}$$

Similarly, the distribution of additional signed observations can be expressed as a real value by merging all deviations Δ_i from the ideal grid defined by S. Then, a set of adjectives is assigned to each of the cues. Those adjectives characterize the real value of their corresponding clue using natural language. An exemplary set of adjectives for the clue VALUE APPROXIMATION is { *perfect*, *good*, *fair*, *poor* }. Note that the adjectives do not need to correspond to crisp intervals of a clue's real value but can gradually fade in and out (cf. Figure 3.13). Next, if-then-statements are used to reason about the correlation of several cues, e. g.:

if DISTRIBUTION is *good* and
  VALUE_APPROXIMATION is (*perfect* or *good*) then
  CONFIDENCE is *high*

It is possible to formulate as many of these rules as necessary for a given application. As shown in the example rule, the CONFIDENCE property is assigned adjectives in the same way as all the cues. Also, the resulting confidence value (in the range of 0–100%) is mapped to the adjectives *low*, *medium*, and *high*. Thus, only a mapping between the input values' adjectives and the corresponding confidence adjectives needs to be expressed by the rules. It is not necessary to express exact correlations between the underlying real values. The resulting confidence percentage is then calculated by evaluating the rules, considering, for example, which confidence adjective has been assigned to the most.

Other components can use confidence values.

Then, all further components of the underlying aggregation mechanism can use this single confidence value to judge the correctness of the aggregate, e. g.:

- If the confidence value is very low, the aggregate can be discarded.
- Fusion of aggregates with highly differing confidence values can be prevented.

— Aggregates with high confidence value can be prioritized for further dissemination.

However, as soon as an aggregate is selected for further dissemination, the assigned confidence value loses its significance. Any vehicle receiving the aggregate will re-evaluate the confidence using the attestation meta-data. Using confidence evaluations only locally eliminates the need for additional security mechanisms that would otherwise be needed to protect the integrity of confidence values during transmission.

Note that the presented confidence rating mechanism may favor information that adheres to a normal traffic model. That is, information where congestions form and dissolve slowly. Information that does not adhere to the model, such as accidents that lead to sudden changes in velocity, may falsely be regarded as attacks. Therefore, an aggregation mechanism should always be complemented with a non-aggregating safety protocol, which disseminates warnings about such rare events without aggregation, and which is protected with other security mechanisms that are outside the scope of this thesis.

The combination of the presented selective attestation mechanism with the confidence fusion allows to combine cryptographic signatures used as trust anchors with probabilistic integrity criteria. The mechanism allows to express arbitrarily complex correlations of those criteria. But due to the fusion into a single confidence value, only minor modifications to the underlying aggregation scheme are necessary to make use of the added security mechanisms.

7.4 SECURITY EVALUATION

For evaluation, we simulate our proposed security mechanisms and compare them against a baseline. As baseline, we use the same underlying aggregation mechanism with no security features. A total of 30 nodes, among which 10 are attackers, move on a 1.5 km highway segment with three lanes. We choose a high fraction of attackers to demonstrate the resilience of our mechanism even when facing a large number of attackers. After 900 m, the highway is blocked by an obstacle, resulting in a developing traffic jam. The size of dissemination messages is fixed. Aggregated information and attestation atomic reports are added according to the rules specified in Section 7.2. For this example application, we chose the security parameter $S = 333\,\text{m}$, i. e., 3 attestations per kilometer are required in addition to the atomic reports at the borders of an aggregate. The choice of S corresponds to the settings of the underlying aggregation mechanism, which chooses aggregates with at most 333 m geographic distance for aggregation. For the temporal dimension, we do not apply S in the simulation, because outdated reports are already ignored by the aggregation mechanism. To assess the effectiveness of the proposed mechanisms, both the achieved security and the induced bandwidth consumption are evaluated.

Security evaluation.

The mean aggregated view of the situation, without an active adversary, is shown in Figure 7.3 (*no attack*). The simulated adversaries now try to conceal the congestion by crafting aggregates that pretend normal traffic flow on the

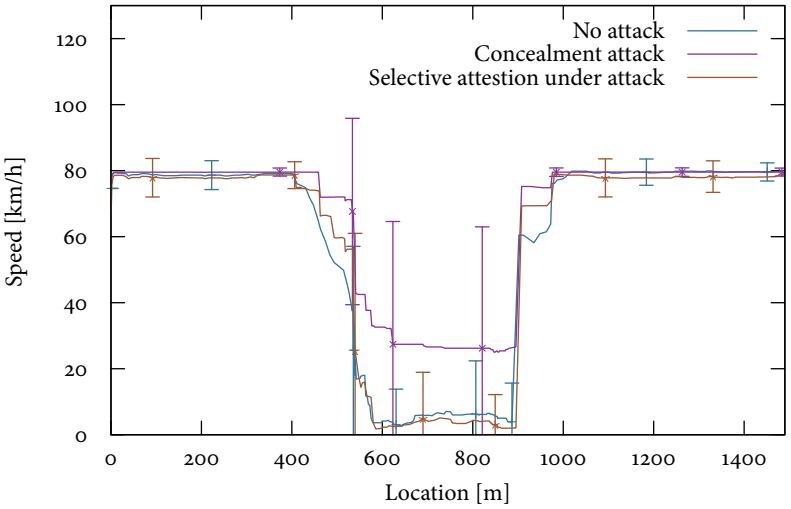


Figure 7.3: Averaged traffic flow in three different situations: the actual situation with no attack, a congestion concealment attack, and the same attack with selective attestation security.

whole road. Without any security countermeasures, the attackers can alter the aggregated view of the situation on average by 25 km/h (CONCEAL). Moreover, under attack, the aggregated views of the vehicles differ notably, shown by a high standard deviation in the graph. With activated security mechanisms, the resulting mean aggregated view is close to the reference view without attack. An attacker trying to conceal the congestion is not able to present enough supporting information. This results in lower confidence in attacker aggregates leading to a correct representation of the real situation.

Scalability evaluation.

Despite the security gain, one important factor of secure aggregation mechanisms is their scalability. To gauge this performance, we use the dissemination speed as a metric, that is, the amount of time needed until all vehicles have information about a high percentage of the upcoming road. This metric implicitly includes the bandwidth overhead caused by the security mechanisms, because the dissemination packet size is fixed. Thus, information dissemination is slower if fewer aggregates fit in one packet. We compare the selective attestation with the performance of the underlying aggregation mechanism without any security considerations and against the periodic dissemination of cryptographically-signed atomic reports without any aggregation.

Figure 7.4 shows that the aggregation without security countermeasures outperforms both security-aware protocols, as expected. The beaconing of atomic reports without aggregation achieves a certain awareness of the vehicles' neighborhood in the first 15 s of the simulation. But then awareness only increases linearly as the vehicles explore more of the simulated road by driving along

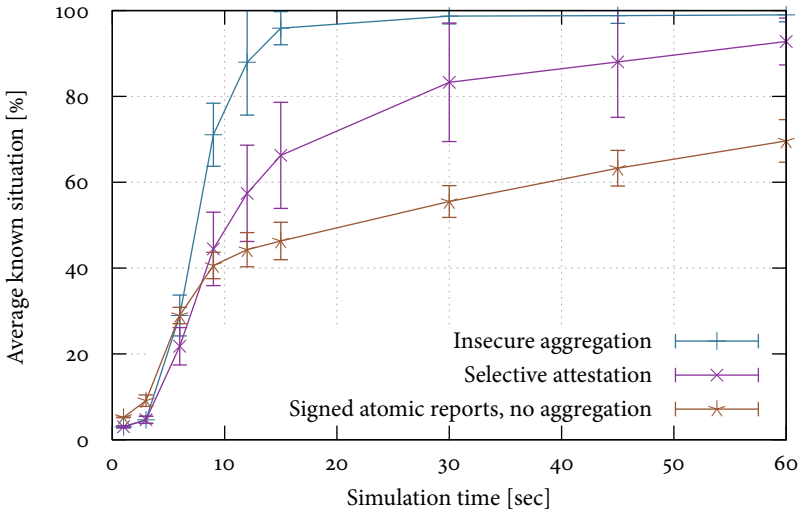


Figure 7.4: Comparison of dissemination speed between the underlying aggregation mechanism without security mechanisms, the selective attestation mechanism, and the dissemination of signed atomic reports without aggregation.

and gathering local sensor values. The selective attestation approach significantly outperforms the dissemination of atomic reports. Although the dissemination speed is slower than the insecure aggregation due to the size of added signatures, at the end of the simulation, over 90 percent of the road is known, whereas the atomic report beaconing only covers 70 percent.

It should be noted that the bandwidth overhead induced by the selective attestation correlates only with the area that is covered by aggregates and is independent of the amount of vehicles in a certain area. Especially in scenarios with very high vehicle density, e. g., traffic jams, the selective attestation therefore clearly outperforms the dissemination of atomic signed reports. Moreover, the computational overhead for the verification of signatures is decreased notably, since only a small amount of atomic reports are transferred, compared to the approach that does not aggregate, i. e., that sends all information as signed atomic reports.

7.5 BANDWIDTH OVERHEAD ANALYSIS

To further analyze the scalability of the selective attestation mechanism, we analyzed its scalability in large areas using a theoretical model. We also propose extensions and mechanisms to compress attestation meta-data using novel cryptographic mechanisms.

Table 7.1: Attestation meta-data size calculation

Component	Description	Size (bytes)
$o_i = ((p,t),v)$	p, t, and v are stored as floats with 4 bytes precision.	12
$\sigma_{v_i}(o_i)$	ECDSA 265 bit signature.	64
pk_{v_i}	ECDSA 265 bit public key.	32
$cert_{v_i} = (T, \sigma_{AA}(pk_{v_i}, T))$	At least, T encodes the validity period using two floats.	72
Total		180

In its basic form, each element of signed attestation meta-data has the following form:

$$(o_i, \sigma_{v_i}(o), pk_{v_i}, cert_{v_i}). \tag{7.6}$$

We call these elements *witnesses*. Table 7.1 shows the size of each contained component, assuming a minimal set of attributes – that is, only the validity period – is used in the certificates. Figure 7.6 visualizes the respective proportions. Each attached witness consumes at least 180 bytes. The total number of witnesses depends on the aggregate locator area's size and the security parameter S. Let $x = b - a$ be the length of the aggregate's geographical area and $y = e - s$ be the time interval covered by the aggregate. Then, according to the selection criteria in Equation (7.3) and Algorithm 7, the total overhead due to selective attestation is at most

$$(4 + \lfloor x/S \rfloor + \lfloor y/S \rfloor) \cdot 180\,\text{bytes} \in O(x \cdot y). \tag{7.7}$$

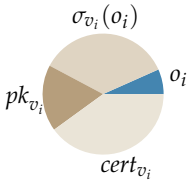


Fig. 7.6: Proportional distribution of overhead.

Neglecting time, the security overhead is essentially linear in the geographical interval covered by an aggregate. Clearly, this contradicts the scalability requirements discussed in Section 3.4.1, namely, the required bandwidth reduction of $o(1/d^2)$ where d is the distance to an event. But that formula assumes that information is forwarded in an infinite dissemination region.

Achievable area coverage.

For practical considerations, we use the MTU as guideline, as discussed in Section 6.3.3. In the presented scheme, the maximum area for which attestation meta-data can be encoded in a single packet is

$$\frac{1500\,\text{bytes}}{180\,\text{bytes}} \cdot 500\,\text{m} = 4000\,\text{m}. \tag{7.8}$$

Hence, the size of a single event reported using the attestation meta data can be *at most* 4 km. If two events need to be disseminated at the same time, each can span at most 2 km, and so forth. Note that this example considers the best case, neglecting time and partial aggregates about the same event that would need to be disseminated in parallel.

To lower the bandwidth overhead, two options are conceivable: 1. the size of a single witness can be reduced, increasing the achievable total event size or

2. the security parameter S can be made adaptive, ideally such that it depends inversely on the aggregate's geographic extent x, making the security overhead constant. We will discuss strategies for achieving option 1 in the following section and strategies for option 2 in Section 7.7.

7.6 META-DATA COMPRESSION

The original scheme considers time and geographical location as equal parameters that need to be protected in the same way. By doing so, the scheme becomes generic. For instance, location information could easily be extended to 2 dimensions. Then, the aggregate locator becomes 3-dimensional, but it can be secured using the same witness selection concept by extending its grid to three dimensions in the obvious way. However, a less generic witness selection approach can result in better bandwidth overhead.

As a first step towards witness compression, we therefore choose different witness selection strategies for time and geographical location. Namely, we

- keep the S-defined grid for the geographical location but
- choose the *newest* available witnesses for the time axis.

The reason for doing so is that most applications require vehicles to have up-to-date information, while few applications require historic information about older time intervals. Therefore, it may suffice to protect integrity of current information. The bandwidth overhead of the modified witness selection is in $O(x)$ instead of $O(x \cdot y)$. If we also omit the witnesses at the time-axis borders, the overhead becomes

$$(2 + \lfloor x/S \rfloor) \cdot 180\,\text{bytes}. \tag{7.9}$$

As evident from Figure 7.6, the main overhead introduced by attestation meta-data is due to the attached signatures, public keys, and certificates. Aggregate signatures [34], which we discussed in Section 4.6.1, allow to reduce the overhead considerably. Identity-based aggregate signatures (IBAS) [189, 73] allow for an even more compact representation of witnesses.

Therefore, we propose two witness compression strategies based on aggregate signatures and IBAS, respectively, in the following. In Section 7.6.1, we show which components of the witnesses are compressible using aggregate signatures. Section 7.6.2 extends the compression idea using IBAS. The main advantages of using identity-based cryptography is its more compact representation of public keys, but we also propose a mechanism to merge keys during hierarchical aggregation that enables dynamic adaptation of the scheme's security parameter S, which we will discuss in Section 7.7.

7.6.1 *Witness compression using aggregate signatures.*

As shown in Figure 7.6, witnesses are composed of atomic observations, signatures, and public keys. Observations are the component with the smallest rep-

resentation. Distinct public keys are an intrinsic requirement of the security mechanism, because they represent agreement of distinct entities with the aggregated values. Therefore, we focus on compressing attached signatures. Each witness contains two signatures: the creating vehicle's signature on the message and the certificate authority's signature on the public key. Therefore, signatures amount for the major part of the security overhead. We now propose a compression approach for these signatures using aggregate signatures.

Let $\{w_1, \ldots, w_n, Q\}$ be the attestation meta-data for an aggregate. Each witness w_i is comprised of an atomic observation o_i and the necessary information to prove that vehicle v_i created w_i where $v_i \neq v_j \forall i \neq j$. In addition, Q represents information that is required to verify the witness list as a whole. In the basic scheme, w_i is defined as in Equation (7.6), and $Q = \emptyset$.

Because the underlying aggregation scheme is DYN, all atomic observations are created independently, and are different from each other. As discussed in Section 4.6.1, we can use aggregate signatures to represent all vehicles' signatures on their observations using a single value. Hence, the meta-data format becomes

$$w_i = (o_i, pk_{v_i}, cert_{v_i}) \tag{7.10}$$

and

$$Q = \sigma_{v_1,\ldots,v_n}(o_1 \vee \cdots \vee o_n) := \prod_{i=1}^{n} \sigma_{v_i}(o_i). \tag{7.11}$$

In addition to the vehicles' signatures, the authorization authority's signature on each certificate can be compressed using an aggregate signature, further reducing overhead. The modified meta-data format is then

$$w_i = (o_i, pk_{v_i}, T_i) \tag{7.12}$$

Recall that T_i represents any additional certificate attributes besides the subject's public key. It contains at least the certificate validity period.

and

$$Q = \{\sigma_{v_1,\ldots,v_n}(o_1 \vee \cdots \vee o_n), \sigma_{AA}((pk_1, T_1) \vee \cdots \vee (pk_n, T_n))\}. \tag{7.13}$$

The aggregate signature can be calculated incrementally by each vehicle that participates in the aggregation process with two restrictions. First, witnesses that have been included in the aggregate signature cannot be removed afterwards. This is not a problem if a fixed S is chosen. However, dynamic values for S might require vehicles to remove witnesses during hierarchical aggregation that were added earlier in the aggregation process. This concept cannot be reflected by the aggregate signature. Second, if overlapping aggregates are further merged and they have overlapping sets of witnesses, it needs to be stored how often each public key is contained. Otherwise, the correctness of the aggregate signature cannot be verified.

The result of using two aggregate signatures instead of two lists of signatures is a considerably lower security overhead. The size of each aggregate signature

Most notably, a GDH group is required, which limits the number of suitable elliptic curves.

further reduces to one curve point instead of two, because of the different underlying group properties [35]. However, the curve size increases to 259 bit due to the required GDH property. [35]. Thus, each aggregate signature consumes 32.375 bytes compared to the 64 bytes consumed by ECDSA signatures. Overall, the overhead becomes

$$(2 + \lfloor x/S \rfloor) \cdot 52.375 + 2 \cdot 32.375 \,\text{bytes}. \tag{7.14}$$

Specifically, a bilinear pairing is used on the GDH group to multiply public keys, exploiting that the pairing solves the CDH problem in the group to check signature validity.

The major remaining linear factor is the size of each participating vehicle's public key. During verification of aggregate signatures, public keys are multiplied in a way similar to the creation of the aggregate signature itself. But applying this multiplication during the aggregation process would break the scheme for two reasons. First, public keys need to be mutually different to ensure that n different vehicles have created the selected witnesses. If public keys were compressed, an attacker could create all n signatures using the same public key without it being noticed. Second, compressing the public key list by multiplication would break the authorization authority's aggregate signature on the keys.

7.6.2 *Witness compression using identity-based aggregate signatures.*

In the previous compression approach, we identified two factors that limit compressibility of public keys: public keys are group elements that have a fixed size, and the certificate authority individually signs each public key. In the following, we present an improved compression scheme based on IBAS [189, 73]. Identity-based cryptography solves both drawbacks of the previous mechanism. Public keys are derived from identities rather than group elements, which reduces their size. Also, certification can be implicit, that is, possession of a valid public key implies certification [147], which enables further adaptive compression techniques for public keys.

Identity-based identifiers.

Intuitively, the public key in identity-based cryptography only encodes the subject's identity, as well as other key attributes, such as the validity period. The authorization authority generates a secret key based on the public key and hands it to the authorized owner. In our case, the key pairs are pseudonymous. Hence, the subject identifier can be a random number r. The complete identity-based public key is then simply a set of values with the form

$$r \,||\, t_1 \,||\, t_2 \tag{7.15}$$

where the time interval $[t_1, t_2]$ identifies the validity period. Note that the public key also acts as certificate, because only the authorization authority (and not the user herself) can calculate correct private keys for a given identity, i. e., public key. To obtain a correct secret key, the user needs to authenticate towards the authorization authority with a valid long-term key. The concept that the possession of the public key implies certification is known as *certificate-less public key cryptography* [147]. The so-obtained key pairs can be used for asymmetric signatures in a way conceptually similar to traditional asymmetric keys. A downside of many identity-based cryptography schemes is that the central

authority knows the secret key of each user. However, keys are only used for signing – and not for encryption – in our case. Therefore, the authority cannot obtain confidential information. What remains is the problem that the authorization authority could create signatures on behalf of the users. While this is a problem, the same is true when using a PKI based on ECDSA. In both cases, the authorization authority can certify arbitrary key pairs and use them for signatures. Therefore, we consider identity-based cryptography as a valid alternative to further compress witness size.

Gentry and Ramzan [189, 73] present an aggregate signature scheme that uses identity-based cryptography based on GDH groups. Using this scheme and the public key format from Equation (7.15), the witness format becomes

$$w_i = (o_i, r_i, T_i) \tag{7.16}$$

and

$$Q = \sigma_{v_1,\ldots,v_n}(o_1 \vee \cdots \vee o_n). \tag{7.17}$$

Flexible selection of the scheme's security parameter.

The identity-based aggregate signatures are represented using two elements from the GDH group. In addition, the scheme requires a random value w that can be used for at most k signatures. The value of w needs to be known to each signer. Following Gentry and Ramzan's argument [73], we assume that loosely synchronized clocks are available to each vehicle and can be used to determine w automatically. Therefore, $\sigma_{v_1,\ldots,v_n}(o_1 \vee \cdots \vee o_n)$ consumes $2 \cdot 32.375 = 64.75$ bytes. The size of each public key is determined by how we choose r. Note that r only needs to be unique within the validity time interval T_i. Moreover, r does not need to be globally unique as long as collisions between participants of the same aggregate signature are rare. Therefore, we argue that a 64 bit – i. e., 8 byte – random value suffices. As a result, the security overhead of the IBAS witnesses becomes

$$(2 + \lfloor x/S \rfloor) \cdot 28 + 64.75\,\text{bytes}. \tag{7.18}$$

The constant overhead is the same as for the ECDSA aggregate signature, because the single identity-based aggregate signatures uses two group elements instead of one. However, the overhead due to the public key list is lowered due to the more compact key representation.

In addition to the smaller public key size, IBAS allow for more flexible selection of the security parameter S. Using regular aggregate signatures, we discussed that only incremental addition of further witnesses is possible. But once signatures are added to the aggregate signature, they cannot be removed. Using IBAS, we can model a function to remove – at least in terms of overhead – witnesses that had already been added to the aggregate signature. To do so, we exploit that Gentry and Ramzan's signature verification algorithm [73] calculates a sum (writing the GDH group additively) based on the signed messages and public keys:

$$\sum_{i=1}^{n} P_{i,0} + \sum_{i=1}^{n} h(o_i, (r_i, T_i), w) \cdot P_{i,1} \tag{7.19}$$

where $P_{i,j}$ are two different hashes of vehicle i's public key, h is a cryptographic hash function, and w is a number based on the loosely synchronized clocks, as explained above. We define a function Φ that performs the addition of public keys and observations in the same way as during signature verification to "remove" witnesses by compressing them to a value of constant size. Contrary to regular aggregate signatures, this kind of public key and message compression is possible, because of the certificate-less cryptography [147], which requires no additional authorization authority signature that could break due to the public key compression.

Example for witness compression.

Consider the following example to see how the witness removal is implemented. Let A_1, A_2 be aggregates accompanied by witness lists $\{w_1, w_2, w_3, Q_1\}$ and $\{w_4, w_5, w_6, Q_2\}$, respectively. Specifically,

$$w_1 = (o_1, r_1, T_1), \tag{7.20}$$
$$w_2 = (o_2, r_2, T_2), \tag{7.21}$$
$$w_3 = (o_3, r_3, T_3), \tag{7.22}$$
$$Q_1 = \sigma_{v_1,v_2,v_3}(o_1 \vee o_2 \vee o_3), \tag{7.23}$$
$$w_4 = (o_4, r_4, T_4), \tag{7.24}$$
$$w_5 = (o_5, r_5, T_5), \tag{7.25}$$
$$w_6 = (o_6, r_6, T_6), \tag{7.26}$$
$$Q_2 = \sigma_{v_4,v_5,v_6}(o_4 \vee o_5 \vee o_6). \tag{7.27}$$

Now, $A_1 \bowtie A_2$ is calculated. We assume that dynamic adaptation of S requires w_2, w_3, w_4 to be removed. Applying Φ as defined above, we now have

$$w_1 = (o_1, r_1, T_1) \tag{7.28}$$
$$w_5 = (o_5, r_5, T_5) \tag{7.29}$$
$$w_6 = (o_6, r_6, T_6) \tag{7.30}$$
$$\begin{aligned} Q_{1,2} = \{&\sigma_{v_1,v_2,v_3,v_4,v_5,v_6}(o_1 \vee o_2 \vee o_3 \vee o_4 \vee o_5 \vee o_6), \\ &\Phi((o_2, r_2, T_2), (o_3, r_3, T_3), (o_4, r_4, T_4))\}. \end{aligned} \tag{7.31}$$

The added Φ output is represented by a single group element and requires 32.375 bytes additional overhead. This overhead, however, remains constant independent of the number of removed witnesses. In the example, only the three remaining witnesses w_1, w_5, w_6 can be checked for correctness. But the aggregate signature verification is still possible, although witnesses were removed at a later point in the aggregation process. Thereby, the operation to remove witnesses offers much more flexibility in selecting the grid size S depending on different kinds of context. Possible algorithms for choosing S dynamically, which make use of witness removal, will be discussed in Section 7.7.

Bandwidth savings summary.

Figure 7.7 summarizes the bandwidth savings due to witness compression. The plot shows the size of attestation meta-data depending on the aggregate's covered geographical area for $S = 500\,\text{m}$. Obviously, each version still exhibits linearly growing overhead. Using regular aggregate signatures, approximately

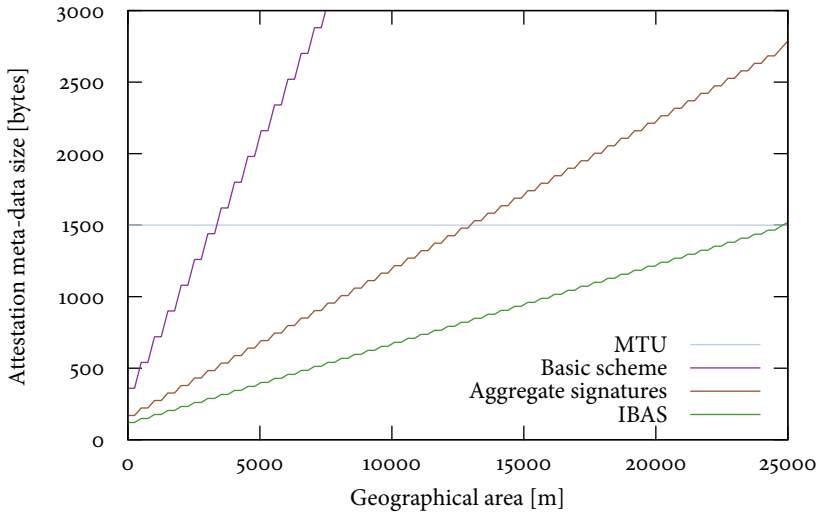


Figure 7.7: Meta-data size comparison for different witness compression methods.

13 km can be covered before a single aggregate exceeds the MTU. Using IBAS, approximately 25 km can be covered. As discussed in Section 7.5, these values consider the overhead of a single aggregate that covers the whole geographical region. However, multiple aggregates representing different traffic situations in the same area would have the same cumulative overhead. Due to overlapping aggregates and duplicate partial aggregates in the network, the achieved coverage will be slightly lower in practice. Still, the achieved coverage suffices for many practical applications.

As discussed in Section 6.3.3, 25 km covered area suffice to inform a driver early enough to take at least one highway exit, which possibly leads to an alternative route. Alternatively, a part of the 25 km stretch can be used to inform about the traffic situation on routes corresponding to the possible exits. In city scenarios, where more alternative routes exist and the road network is denser, 25 km may not suffice to fulfill practical application requirements. In these cases, additional adaptivity of the security mechanism is required, which we will discuss in the following section.

7.7 ADAPTIVITY

In addition to compressing witnesses, the scheme's bandwidth usage can be reduced by adapting parameters that influence the trade-off between achieved security and bandwidth overhead. The witness-based scheme offers one such parameter, namely S, which defines the amount of witnesses added to aggregates. If S is chosen to be

- *small,* more witnesses are added to the aggregate, more bandwidth is used per aggregate area, and it is harder for attackers to create enough witnesses to support false information; if S is chosen
- *large,* fewer witnesses are added, less bandwidth is used, but it is easier for attackers to create false aggregates with correct witnesses.

As an example consider $S = 5000\,\text{m}$. Suppose there is a 2000 m-long traffic jam in between two areas of free-flowing traffic. With the chosen S, an attacker can create a false aggregate that claims free-flowing traffic where there is actually a traffic jam and use witnesses from the surrounding area to support the false information. If S were chosen smaller (e. g., 500 m), the same attack would not be successful. On the downside, the smaller S would increase the bandwidth used by a factor of 10.

The basic scheme presented in Section 7.2 uses a constant value for S, which is chosen at design time. In the following, we will discuss options for choosing S dynamically. Recall that the IBAS-based witness compression supports dynamic values for S while preserving the bandwidth enhancements over the basic scheme. The goal is to choose S such that little bandwidth is used, but the statistical chances of a successful attack are low.

There are several levels of adaptivity that can be applied to S.

SYSTEM The system level offers no adaptivity. S is chosen constant as in the basic scheme.

STATIC CONTEXT Static context means all kinds of context that can be determined at design time, but varies according to time or geographical location. For instance, S could be chosen larger for highways and smaller for city scenarios, as proposed by Molina-Gil et al. [124].

DYNAMIC CONTEXT In addition to static context, dynamic context means all environmental influences that vary during runtime or are learned based on previous interactions. For instance, if many spurious messages are received form the same geographical region and few from another, S decremented for that region.

AGGREGATE PROPERTIES All values contained in the aggregate can be used to choose S. For instance, larger S values can be chosen for aggregates covering larger geographic regions.

AGGREGATE CONTEXT Aggregate context refers to all information that has been used to create an aggregate A, that is, all information in $P^{\bowtie}(A)$. For instance, S can be chosen smaller if the values in $P^{\bowtie}(A)$ exhibit a high standard deviation ς.

These levels have different underlying trade-offs. The system context offers a constant security protection with known probabilities of an attacker succeeding. Moreover, the bandwidth usage can be calculated, and S can be chosen to support a certain aggregate area, as discussed in Section 7.6. Static context is a

straight-forward extension, which allows to adapt the system to different environments. Both of these adaptivity levels result in a choice of S that is constant throughout an aggregate's lifetime, assuming that the context used is coarse (e. g., city vs. highway). That is, given an aggregate A, each partial aggregate that was used in A's creation, as well as each aggregate that may be created based on A and other aggregates will use the same value for S.

All other adaptivity levels may result in differing values for S during an aggregate's lifetime. As a consequence, all of these levels require a mechanism to remove witnesses that were once added to the aggregate but are no longer required. Consider the following example. Let

$$S = \begin{cases} 300\,\mathrm{m}, & |b-a| < 1000\,\mathrm{m}, \\ 600\,\mathrm{m}, & 1000\,\mathrm{m} \leq |b-a| < 3000\,\mathrm{m}, \\ 900\,\mathrm{m}, & \text{otherwise.} \end{cases} \tag{7.32}$$

Here, a and b denote the aggregate's location extent; see Equations (7.1) and (7.2).

That is, S is selected depending on the aggregate geographical area. Further, let $o_0, \ldots, o_9$ be observations with the following values and let w_i be the witness containing o_i.

$$\begin{aligned} o_0 &:= ((0\,\mathrm{m}, 0\,\mathrm{s}), 50\,\mathrm{km/h}), \\ o_1 &:= ((100\,\mathrm{m}, 10\,\mathrm{s}), 50\,\mathrm{km/h}), \\ o_2 &:= ((200\,\mathrm{m}, 20\,\mathrm{s}), 50\,\mathrm{km/h}), \\ &\vdots \\ o_{10} &:= ((1000\,\mathrm{m}, 90\,\mathrm{s}), 50\,\mathrm{km/h}). \end{aligned} \tag{7.33}$$

Now suppose $A_1 = o_0 \bowtie \ldots \bowtie o_6$ is calculated. The aggregate area is $|b-a| = 600\,\mathrm{m}$. Therefore, witnesses w_0, w_3, w_6 are selected as meta-data. Then A_1 is disseminated further and a different vehicle calculates $A_2 = A_1 \bowtie o_7 \bowtie o_8 \bowtie o_9 \bowtie o_{10}$. Because of the new aggregate area, $S = 600\,\mathrm{m}$ and witnesses w_0, w_6, w_{10} are selected. The previously added w_3 is discarded. For the basic scheme, such removal of witnesses is trivial. For the optimized scheme using meta-data compression, we presented a method to remove witnesses using IBAS in Section 7.6. In both cases, note that the adaptation of S must take into account that aggregating vehicles do not have access to all possible witnesses. For instance, the vehicle creating A_2 cannot access o_1, o_2, o_4, and o_5 anymore.

The previous example clearly saves bandwidth for larger aggregates, but it may open new attack vectors due to the larger choices for S. In particular, the mechanism makes it relatively easy for an attacker to forge large aggregates, which in turn may have larger impact on other vehicles. Thus, the example adaptation mechanism favors dissemination area over integrity protection. In general, we can group adaptation mechanisms into those that focus on dissemination area maximization and those that focus on integrity protection as the primary optimization metric. The respective other goal is then used as a secondary metric. We argue that using integrity protection as the primary metric

should be favored because area-centric mechanisms may lead to unpredictable attack vectors as discussed above.

Area maximization adaptivity, such as in the example discussed above, can be applied by the aggregating vehicles and independently verified by receiving vehicles without additional information. For instance, each receiving vehicle can verify whether the attached amount of meta-data corresponds to the aggregate area. For integrity-centric mechanisms, received aggregates may not suffice to verify the meta-data selection. Consider the following two examples.

1. To use *dynamic context,* such as previous attacks, vehicles need to create and maintain a database of previous attacks associated with different parts of the road network. Further, the database should be loosely synchronized between aggregating and receiving vehicles so that both expect the same values for S. Possibly, additional messages need to be exchanged to achieve synchronization.

2. To use *aggregate context,* such as standard deviation of used values, the standard deviation within the aggregate area needs to be independently verified by the receiving vehicle. The standard deviation ς added by the aggregating vehicle may have been forged. Verification is only possible if the receiving vehicle has access to other redundant information about the aggregate area, such as atomic observations from other vehicles or aggregates created from different vehicles.

As can be seen, information from the aggregation protocol needs to be augmented with other historical information and redundant information. The mechanisms to gather and evaluate such additional information may be orthogonal to the integrity mechanism itself, and it may itself be subject to attacks. We therefore argue that adaptation of the integrity mechanism should not be an intrinsic part of the mechanism itself. Rather the mechanism should offer suitable parameters, such as the witness granularity S. The configuration of these parameters should then be performed by a generic framework, which takes into account several security mechanisms and their interrelations. We present such a framework in Chapter 11, which we will use to evaluate the adaptation strategies we proposed in this section.

7.8 SUMMARY

We have presented a basic scheme for integrity protection of in-network aggregation. To detect attacks, additional meta-data in the form of witnesses is added to each aggregate. A security parameter S ensures that enough meta-data is selected to thwart attacks while incurring minimal bandwidth overhead. We evaluate the basic scheme's integrity protection against the attacker model presented in Section 4.4.3 and present enhancements to decrease bandwidth overhead. An IBAS-based signature scheme offers best witness compression while preserving the flexibility to dynamically adapt the security parameter S.

Finally, we discussed adaptivity strategies for S and motivate the need for a generic framework to combine and adapt mechanisms, rather than implementing adaptivity separate for each security mechanism. The presented scheme can be used in different traffic situations, but its overhead is linear in the geographic area covered by aggregates. For very large areas, the overhead may be too high despite the introduced compression mechanisms.

8

This chapter is a revised and extended version of our publication [14]. Preliminary investigations for this chapter were conducted as part of a supervised master thesis [209].

8.1 OVERVIEW

The scheme we presented in Chapter 7 provides protection against aggregate manipulations. However, applying the optimized scheme presented in Section 7.6, the bandwidth overhead still grows linear in the aggregate area for a fixed aggregation granularity S. In the basic scheme, the overhead is linear in the geographic area, as well as in the timespan represented by an aggregate. Another problem is that the scheme is sensitive to duplicate information. That is, multiple reports about the same event may bias aggregation. Here, we present a protection mechanism that performs better in these two aspects.

The underlying aggregation mechanism is based on DYN with two notable changes. First, we observe that traffic information on a highway is more relevant for approaching upstream vehicles than for downstream vehicles. Therefore, the scheme aims to disseminate information predominantly upstream. Second, we introduce an agreement phase where vehicles determine the extent and characteristics of an event, which we separate from the phase where the event information is verified, attested, and disseminated. This separation allows to use more lightweight cryptography.

We employ a three-phase approach for combining flexible aggregation with cryptographic integrity protection, as shown in Figure 8.1. First, vehicles disseminate atomic observations about their current position, time and speed. Observations are forwarded and aggregated with other downstream observations until the edge of a homogeneous traffic stretch is detected. We use a local algorithm to detect these edges: each vehicle examines its neighborhood for changes in velocity. In Figure 8.1, the blue area marks the homogeneous area, and $\mathcal{A}$ is the aggregate containing information about the whole area. In this phase, integrity-protected FM sketches are used for protection.

In the second phase, the preliminary aggregate $\mathcal{A}$ is disseminated again to all participating vehicles, this time upstream. In this phase, each vehicle signs the preliminary aggregate to attest its correctness, as shown by the σ_i's in the figure. Finally, a third phase is used to disseminate the finalized, signed aggregate $\mathfrak{A}$ further upstream to other vehicles that are not driving within the aggregation area. As aggregates do not change anymore during the finalization and dissemination phases, multi-signatures [31] and identity-based cryptography [50] are used to keep security overhead low. For instance, the final aggregate $\mathfrak{A}$ only carries a multi-signature $\sigma_1 \circ \cdots \circ \sigma_4$ instead of the 4 individual signatures. In the following, we describe each phase in more detail.

The scheme focuses mainly on aggregation of traffic information. An adaptation to, for instance, parking spot aggregation, is conceivable but requires adaptation of the scheme. Specifically, the first phase would need to be modi-

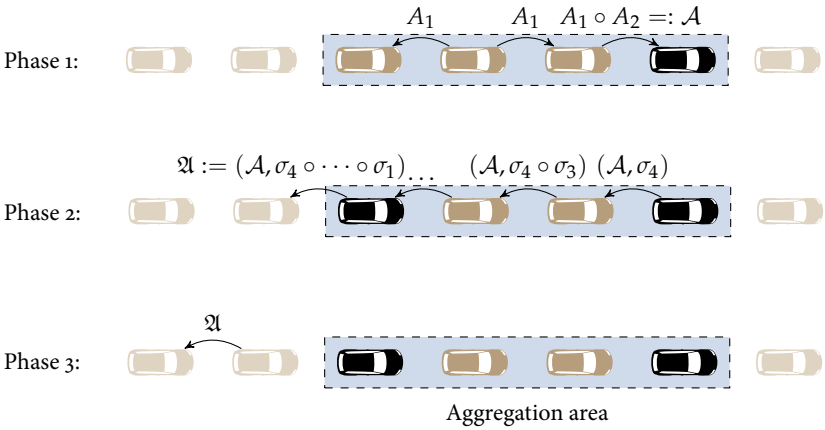


Figure 8.1: Overview of our three-phase security approach.

fied, whereas phases two and three could be re-used and benefit from the multi-signatures' low overhead.

8.2 AGGREGATION PHASE

The goal of the aggregation phase is to collaboratively determine the average velocity within a stretch of road with homogeneous traffic, as well as the road stretch's extent. To do so, we present a data representation format that allows for duplicate-insensitive information fusion, a communication protocol, and an integrity protection mechanism. Note that the aggregation mechanism is only used by vehicles within the same traffic situation. Once the aggregated information reaches the border of the traffic situation, the aggregate enters the finalization phase.

Using probabilistic sketches.

To represent information, we use two kinds of probabilistic sketches. The major advantage of sketches is their duplicate insensitivity, which comes at the expense of probabilistic results [114]. So no matter how often a vehicle contributes to an aggregate, its values will only be counted once, and no extra storage is required to achieve this duplicate insensitivity.

Using an `add()` operation, sketches can directly represents counts of distinct values. Sums of arbitrary integer values can be represented by concatenating n `add()` operations to add the value n. An `evaluate()` operation is used to extract the estimated number of distinct elements from a sketch. An average can be represented by dividing a sum sketch by a corresponding count sketch.

In our case, information is represented using two averages: one representing the center position of the aggregate area – that is, the average of all contained vehicle positions relative to a fixed point – and one representing the average velocity. For the sum sketches, we use FM sketches, which have been applied by a number of aggregation mechanisms discussed in Section 3.5. Moreover,

Algorithm 8: FM sketch functions

DATA: Let $S_i = s_{i,1}, ..., s_{i,w}$ be a bit field of size w and let $S_1, \ldots, S_m$ be the set of bit fields to use.
DATA: Let h be a *geometrically distributed* hash function with positive integer output, that is, $P(h(x) = i) = 2^{-i}$.

```
FUNCTION init()
    FOR 0 < i ≤ m DO
        FOR 0 < j ≤ w DO
            s_{i,j} ← 0
        END
    END
END
FUNCTION add(x)
    i ∈_R {1,...,m}                          /* sketch to add to */
    j ← h(x)                                 /* position in sketch */
    s_{i,j} ← 1
END
FUNCTION estimate()
    ρ ← 0.775351
    Z ← 0
    FOR 0 < i ≤ m DO      /* find S_i's longest 1-bit sequence */
        Z ← Z + min({j ∈ ℕ_0 : j < w ∧ s_{i,j+1} = 0} ∪ {w})
    END
    RETURN (m/ρ) · 2^{Z/m}
END
```

several approaches for securing FM sketches already exist and can be adapted (see Chapter 5). To increase FM sketch accuracy, we use probabilistic counting with statistical averaging (PCSA), as suggested in Flajolet and Martin's paper [70]. For counts, we use linear counting sketches (LC-sketches) [167], because they offer better accuracy for a small number of added elements, as argued by Fan and Chen [63].

We have already discussed the FM sketches data structure in Section 3.5; we summarize the main functions in Algorithm 8 based on [114]. LC-sketches are similar in that they also use a bit array of size $w \ll n$ to estimate the number of distinct elements in a stream of length n. Algorithm 9 summarizes the LC-sketch functions.

Description of complete data structure.

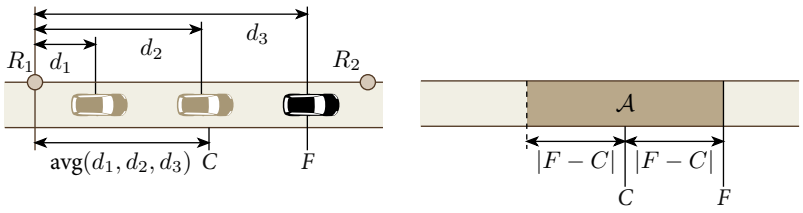
Our complete data structure works as follows. Whenever a vehicle contributes to an aggregate, it adds its identity to an LC-sketch representing the count of the aggregate's contributors c. In addition it adds its velocity to an FM sketch v. The average aggregate velocity can then be estimated as v/c.

The use of sketches to represent the aggregate's geographical area is more complex, but it is required to achieve duplicate insensitivity and to be able to apply the same integrity protection mechanism to both the average velocity and the geographical area. Our goal is to represent the aggregate area by the average

Algorithm 9: LC-sketch functions

DATA: Let $S = s_1, ..., s_w$ be a bit field of size w.
DATA: Let h be a *uniformly distributed* hash function with positive integer output.

```
FUNCTION init()
    FOR 0 < i ≤ w DO
        s_i ← 0
    END
END
FUNCTION add(x)
    i ← h(x)                                  /* position in sketch */
    s_i ← 1
END
FUNCTION estimate()
    Z ← {0 < i ≤ w : s_i = 0}                 /* number of 0-bits */
    RETURN −w ln |Z|/w
END
```



(a) Calculation of aggregate dimensions.

(b) Interpolation of aggregate area.

Figure 8.2: Calculation and representation of the aggregate area. First, vehicles encode their relative distances d_i to a reference point R_1. The aggregate area is then estimated by the approximate center C, which is determined as the d_i's average, and the finalizing vehicle position F.

position of all vehicles contributing to the aggregate. To represent vehicle positions as small integer values, we introduce a set of reference points $R_1, \ldots, R_n$ along the street. The reference points are fixed and can be derived by each vehicle knowing the distance between two reference points and having a map of the road network. Each vehicle then encodes its position as relative distance d_i to the last reference point it passed, as shown in Figure 8.2a. The d_i values are added to an FM sketch d. The aggregate center C can then be approximated as d/c.

Leading over to finalization phase.

The aggregation process continues downstream until the end of a homogeneous stretch of traffic is encountered. Each receiving vehicle compares the known vehicle speeds in its direct neighborhood to evaluate whether it is at the border of a homogeneous stretch. Once a border is reached, the position F of the vehicle that detected the border – the so-called *finalizing vehicle* – is added to the aggregate. The aggregate center C together with F can be used to

estimate the total area covered by the aggregate, as shown in Figure 8.2b. The underlying assumption is that vehicles within a homogeneous stretch of traffic are uniformly distributed. Then, the distance between the center C and F estimates the aggregate area's radius, and $2 \cdot |F - C|$ is an estimation of the aggregate's diameter.

During the aggregation phase, information is forwarded predominantly downstream. Each vehicle broadcasts its current velocity and position using sketches as described above. Receiving vehicles add their own position and speed to the sketch and re-broadcast the packet if they are further downstream. To avoid collisions, each vehicle uses a random delay before broadcasting. Once the aggregate reaches the finalizing vehicle, the aggregation phase ends. At this point, vehicles within a stretch of homogeneous traffic have determined their average velocity and the road stretch extent.

Integrity protection for sketches.

We use an integrity protection mechanism similar to those proposed by Han et al. [78] and Garofalakis et al. [72]. All data structures we use are based on bit arrays, which are initially filled with 0s and subsequently set to 1 according to different algorithms. Despite possible hash collisions, the majority of 1-bits can be expected to be set by mutually different vehicles. Therefore, each vehicle that sets a previously 0 bit to 1 in a sketch adds a signature to protect the sketch integrity. The signature contains the reference point R_i, the bit's position in the sketch, as well as the current time period. Adding the reference point ensures that signatures cannot be re-used by other vehicles from different location areas. Similarly, the time period ensures freshness of the signatures. To save bandwidth, IBAS (cf. Section 7.6) can be used instead of regular ECDSA signatures.

Note that certain attacks on this protection scheme are possible. To represent the sums using FM sketches, each vehicle adds multiple values to a sketch. Therefore, some signatures in one sketch may be created by the same vehicle. An attacker could potentially exploit this and falsely set bits in a sketch to 1 to inflate the sketch value. To hinder this form of attack, receiving vehicles can use probabilistic approaches to verify value plausibility. For instance, aggregates with unrealistic velocity averages can be discarded. In addition, it is statistically unlikely that the majority of signatures in a benign sketch originate from the same signer. Such aggregates can be discarded, too.

Note that the bandwidth overhead of these signatures is significant. Each aggregate is represented by a total of 3 sketches: 1. sum of vehicle positions, 2. sum of vehicle velocities, and 3. count of vehicles. Within each sketch, the number of signed 1-bits rises as more vehicles participate in the aggregate. However, the sketch-based integrity protection is only used in the aggregation phase. Because such messages are only exchanged amongst vehicles within the same traffic situation, only one aggregate must be disseminated at a time. Moreover, the maximum amount of signatures is bound by the cumulative size of all sketches. For further dissemination, where more than one aggregate about several different traffic situations is disseminated, we will use a different, light-weight protection mechanism.

8.3 FINALIZATION PHASE

Once the aggregation phase is ended by the finalizing vehicle, the resulting aggregate should represent a common view of the traffic situation area and average velocity. The goal of the finalization phase is to re-disseminate the aggregate throughout the aggregation area and perform an agreement protocol. The agreement protocol takes the integrity-protected sketches from the aggregation phase as input and replaces it by a lightweight integrity protection. We first describe the integrity protection conceptually. Then we discuss an extension that eliminates problems introduced by redundant broadcast dissemination.

During the finalization phase, messages are disseminated upstream starting at the finalizing vehicle. Due to the collaborative nature of the protocol, it is possible that multiple finalizing vehicles exist. Before re-broadcasting finalized aggregates, vehicles therefore wait a certain time interval. If multiple finalized aggregates are received during this time, only the finalized aggregate from the vehicle with the lowest id is disseminated further. Each vehicle in the aggregation area that receives the finalized aggregate performs a number of data-consistency checks. Namely, it verifies that the signatures on the sketches are not all created by the same vehicle. In addition, it checks whether the average velocity contained in the aggregate is similar to the own velocity and that of direct neighbors. If all checks pass, the vehicle signs the finalized aggregate and broadcasts it. The list of distinct signers serves to increase confidence in the aggregate's correctness.

To lower overhead, we use an identity-based multi-signature scheme, which Gentry and Ramzan [73] present as a stepping stone towards their IBAS scheme, and which we used for meta-data compression in Section 7.6. Like aggregate signatures, multi-signatures require a list of signer public keys and certificates to be attached to the aggregate for verification. Again, we exploit that identity-based cryptography allows to use short strings as public keys, which serve a dual-purpose as implicit certificates. The key difference to the mechanism discussed in Section 7.6 is that we use multi-signatures here. Multi-signatures require that all signers agree on a common message. In this scheme, the aggregation phase serves to create this message. The advantage of multi-signatures over aggregate signatures is that they require less computation steps. In addition, more vehicles sign their agreement to the message content. In the scheme we presented in Chapter 7, only a subset of all vehicles in the aggregation area are added to the aggregate. Here, all vehicles in the aggregation area can contribute to the multi-signature, which increases confidence in the correctness of the contained information. Multi-signatures can be created incrementally. That is, each vehicle that verified the aggregate content can contribute to the existing multi-signature and attach its public key to the aggregate.

IBAS require a randomness factor w, which can be re-used up to k times, and for larger values of k, computation of signatures is expensive. w is not required for multi-signatures. For details on w's computational impact, see [73].

Once signatures have been added to the multi-signature, however, they cannot be removed without invalidating the signature. The missing removal can be a problem in our broadcast dissemination setting. When a vehicle receives several partial aggregates with overlapping multi-signatures, it cannot calcu-

late a joint multi-signature without producing duplicate signature entries. For example, suppose a vehicle receives two aggregates with multi-signatures:

$$A_1 = (\ldots, (\sigma_1 \cdot \sigma_2), pk_1, pk_2) \tag{8.1}$$

$$A_2 = (\ldots, (\sigma_1 \cdot \sigma_3), pk_1, pk_3). \tag{8.2}$$

If the vehicle would combine both aggregates, it would calculate the multi-signature $(\sigma_1 \cdot \sigma_2 \cdot \sigma_1 \cdot \sigma_3)$, which contains σ_1 twice. For successful verification, pk_1 would then also need to be included twice in the resulting aggregate, consuming unnecessary space. A counter could be used instead to signify how often a public key needs to be used during verification. But an attacker could increase the counter, which constitutes a denial-of-service attack. To solve this issue, we essentially delay the multi-signature calculation by one hop. Each vehicle adds its own signature σ_i separately and integrates σ_{i-1} from its predecessor into the multi-signature. This approach detects and corrects a majority of overlaps that happen in the direct vicinity of vehicles.

At the end of the finalization phase, the finalized aggregate is accompanied with a multi-signature and a list of identities that serve as public keys, which attests to the aggregate correctness.

8.4 DISSEMINATION PHASE

Once the finalized aggregate reaches the opposite end of the traffic situation, it is transformed for further dissemination outside the aggregation area. The sketches used during the aggregation and finalization phase can be discarded and replaced by their estimate values. The multi-signature is kept and serves as integrity protection during further dissemination. The multi-signature is not affected by the changes to the information, because the multi-signature has been created on the aggregate's values (e. g., average velocity) instead of their representation (e. g., the FM sketches). During the dissemination phase, any generic broadcast [e. g., 24] or a secured geocast [e. g., 18] can be used. Receiving vehicles can evaluate the multi-signature to derive confidence in the aggregate's correctness. For instance, the number of signers should correlate with a function of the geographical area and the claimed average velocity.

8.5 SECURITY EVALUATION

We use a highway setting to evaluate the security of our mechanism. Vehicles are initially placed randomly on a 5 km long 2-lane highway. We simulate a low density setting with 200 vehicles and a high density setting with 800 vehicles. Honest vehicles try to maintain a target speed of 90 km/h. We select a random fraction of all vehicles to behave as independent attackers. Simulations are repeated with attacker ratios of 0–10 percent. Attackers implement the FAKE attacker model we introduced in Section 4.4.3 and create messages pertaining to a traffic jam. Table 8.1 summarizes the sketch sizes used in our implementation. Note that we store the vehicle locations with a granularity of

Table 8.1: Sketch sizes used for evaluation.

Use	Granularity	Type	No. sketches	Sketch size	Total size
Participant count	1	LC	1	64 bit	64 bit
Location	10 m	FM	4	8 bit	32 bit
Velocity	1 km/h	FM	8	8 bit	64 bit

10 m to lower the values added to the FM sketch and reduce the required sketch size.

Evaluation scenarios.

We compare four different scenarios to verify our implementation and assess the achieved integrity protection.

MECHANISM 1: The *no security* mechanism is a modified implementation of our scheme that uses the same decision, fusion, and dissemination mechanism but does not validate information or check signatures.

MECHANISM 2: The *full* mechanism implements all security mechanisms we discussed, namely FM sketch protection during the aggregation phase, as well as multi-signatures in the dissemination phase..

MECHANISM 3: The *lightweight* mechanism performs the aggregation phase *without* security mechanisms, but uses the multi-signature agreement discussed for the finalization phase.

MECHANISM 4: The *SAS* mechanism is an improved implementation of Han et al.'s proposal [78], which uses protection mechanisms similar to those we use in the aggregation phase but lacks the finalization phase and its multi-signatures.

All four mechanisms are compared to the *baseline*, which represents the correct real world situation as derived from the mobility model. Calculating the difference between a scenario and the baseline implements the quantification model discussed in Section 6.3.3. We compare against SAS, because of its similar application of FM sketches for protection of aggregates. Moreover, we evaluate a variation of our scheme (Scenario 3), because it offers a different trade-off between bandwidth use and integrity protection. Even when the aggregation phase is not integrity protected, most attacks should be detected during the finalization phase due to the data consistency checks applied.

Figure 8.3 shows the simulation results for different vehicle densities and fractions of attacking vehicles. We vary the amount of attackers from 0 (i. e., no attackers) to 0.1 (i. e., 10 percent attackers), as shown on the x-axes. The y-axes show the average perceived speed according to the aggregated information available to the vehicles. We repeated each simulation 8 times with different random seeds to eliminate statistical effects. The error bars in the graphs show the standard deviation of the averaged values.

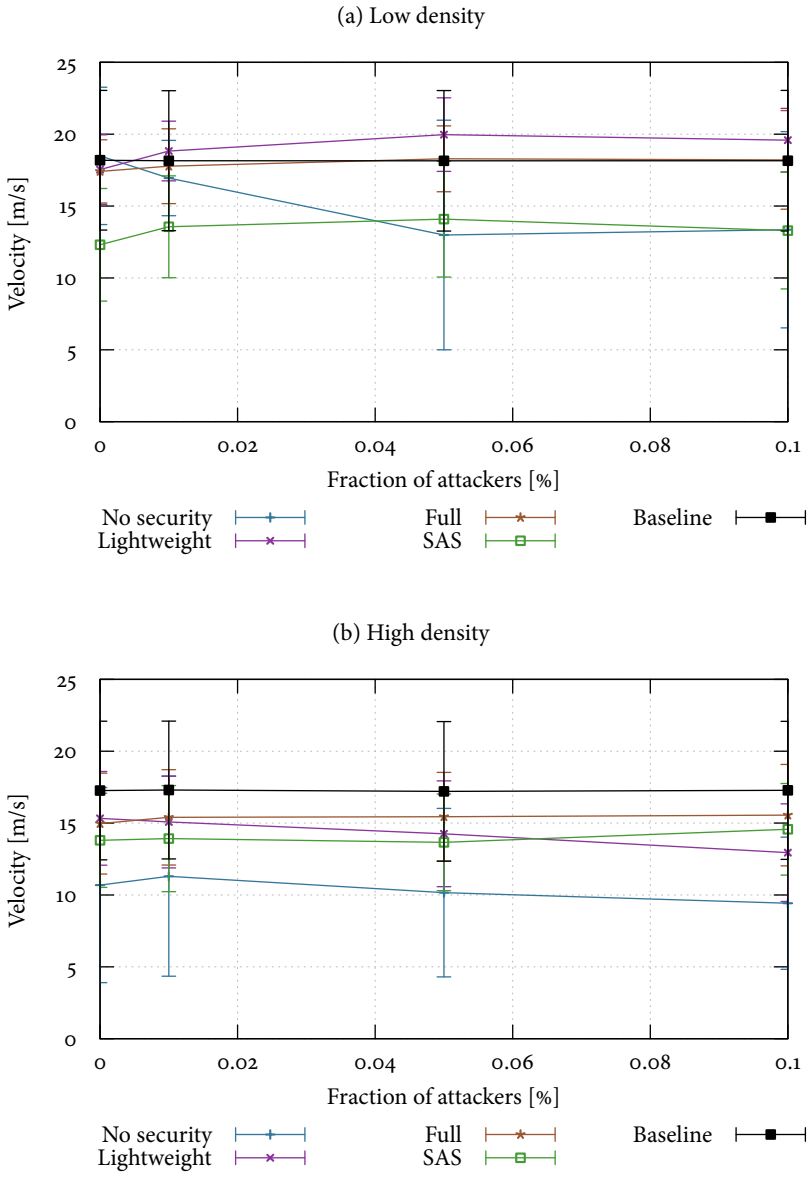


Figure 8.3: Attack impact for different scenarios, different fractions of attackers, and different vehicle densities.

In the low density setting (Figure 8.3a), both the full and the lightweight mechanism perform well and closely represent the baseline despite growing fractions of attackers. The higher standard deviation in face of attacks shows that the attacker has an impact on some vehicles, but on average, the attacker information is correctly filtered. The decreasing average velocity of the no security implementation shows the attacker's impact without countermeasures, which validates the implementation of the attacks. Without security in place, 5 percent attackers can lower the average perceived speed by 15 km/h. The standard deviation increases significantly, which indicates that the attacker is even more successful for some vehicles. Also noteworthy is that the SAS mechanism consistently underestimates the speed, even without any attacker. The reason is that SAS uses the – in this situation – less accurate FM sketches for counting the number of participating vehicles. The simulation results, therefore, validate our choice of LC-sketches as a complementary sketch for vehicle counts.

In the high density setting (Figure 8.3b), the approximation of the baseline is generally less accurate for all mechanisms. This result shows the inherent inaccuracies of the sketches, which increase when more distinct elements are added. Moreover, the higher vehicle density causes more heterogeneity in the traffic situation due to the cars following a mobility model, which is reflected in the simulation results. The heterogeneous traffic especially impacts the no security implementation of our scheme. Here, outliers impact the estimation in the same way as attackers. Both the lightweight and the full security implementations of our scheme successfully compensate these effects, as well as the attacker information. The difference to SAS, however, is lower in the high vehicle density setting.

Successful attack detection.

Note that both the full and the lightweight implementation of our security mechanism successfully detect attacks. In both the low density and high density setting, the full security implementation's estimations are, on average, closer to the baseline. However, differences are not significant. We, therefore, conclude that the lightweight security implementation may suffice especially for aggregates about small and medium size geographic regions, such as those we simulated. For larger areas or more heterogeneous traffic situations, the full implementation, including the aggregation phase security, offers additional protection. In those cases, vehicles participating in the finalization phase cannot determine aggregate correctness purely based on their own sensor values and are more reliant on the signatures added during the aggregation phase.

8.6 BANDWIDTH OVERHEAD ANALYSIS

The mechanism's security overhead is comprised of two major components:

- the aggregate signatures on the various sketches used during the aggregation phase,
- the multi-signatures used during the finalization phase, and
- the requirement of multiple phases before information is disseminated.

We first calculate the total security overhead, which occurs if the full scheme is implemented. We then compare the full overhead to the lightweight variant's overhead. To simplify our model, we neglect transmission delays due to the multiple phases in our calculations. However, we amount for the phases' overhead by factoring in that each information item needs to be represented twice and cumulating the respective overhead in our calculations.

Factors for overhead.

The most overhead of the full scheme is incurred during the finalization phase. Here, the security overhead created during the aggregation phase is still attached to the aggregate, and the multi-signatures are created in addition. Unlike the overhead of the scheme presented in Chapter 7, the overhead of this scheme depends on a number of factors in addition to the aggregate's geographical area. Let

- x be the aggregate's geographical area,
- D be the vehicle density measured in vehicles per meter,
- V be the average velocity, and
- $P(x)$ be the sum of distances to the closest reference point.

All factors but the relative distance $P(x)$ are constant. To approximate $P(x)$, we assume that vehicles are uniformly distributed along the road. Then, there is one vehicle every $1/D$ meter on the road. The first vehicle's relative position is 0 meter, the second position is $1/D$ meter, the third $2/D$ meter, and so forth. In general the sum of positions is

$$P(x) = \sum_{k=1}^{xD} \frac{k}{D} = \frac{1}{D} \sum_{k=1}^{xD} k = \frac{1}{D} \frac{(xD)(xD-1)}{2} = \frac{x^2 D - x}{2}. \tag{8.3}$$

The overhead due to the aggregation phase depends on the number of bits set to 1 in the various sketches, because each 1-bit is signed by the vehicle that first set it. Normally, we use the number of 1-bits in a sketch to approximate the number of distinct elements contained. Here, we invert the sketch approximation formulae to get an approximation for the number of 1-bits dependent on the number of vehicles contributing to the sketch, which in turn depends on the aggregate's area. The approximation formulae, therefore, are different for LC-sketches and FM sketches.

We use LC-sketches to count the contributing vehicles. For an aggregate area of x, the average number of contributors is $x \cdot D$. Applying the LC-sketch estimation formula (see Algorithm 9), we have

$$xD = -w \ln \frac{w - N_c}{w}, \tag{8.4}$$

where N_c is the number of 1-bits in the sketch. Solving Equation (8.4) for N_c yields an approximation formula for the number of 1-bits, i. e., the number of signatures, dependent on the aggregate area:

$$N_c = \max(w - w \cdot \frac{1}{\exp(\frac{xD}{w})}, w). \tag{8.5}$$

For each signature, we use the IBAS scheme introduced in Section 7.6. As stated in Section 8.2, each participant signs the aggregate's reference point (encoded as a floating point value), the bit position in the sketch, and the current time period. Each participant further attaches its public key. In addition, a single aggregate signature is attached, which represents each participant signature. The total security overhead for the LC-sketch is, therefore,

(1) sketch position
(2) public key
(3) reference point
(4) time
(5) aggregate signature

$$O_{\text{count}} = N_c \cdot (\underbrace{1}_{(1)} + \underbrace{16}_{(2)}) + \underbrace{4}_{(3)} + \underbrace{4}_{(4)} + \underbrace{64.75\,\text{bytes}}_{(5)}. \tag{8.6}$$

For encoding relative positions on the road, FM sketches are used. Each vehicle contributes its own position by adding a corresponding number of distinct elements to the sketch. Each vehicle adds a relative position, and the sum of all relative positions added to the sketch is approximated by $P(x)$, as explained in Equation (8.3). The result is divided by the granularity (10 m) with which positions are encoded (see Table 8.1). The FM sketch is evaluated by counting the longest sequence of 1-bits. Analogous to the LC-sketch, we invert the estimation formula to get the number of 1-bits that will be signed. Due to the construction of the hash function, the number of 1-bits not contained in the initial sequence is negligible. Therefore, we have the following approximation for the 1-bit count N_p of the position sketch:

$$N_p = \min\left(m \cdot \log \frac{\frac{P(x)}{10}\rho}{m}, m \cdot w\right). \tag{8.7}$$

Thus, the overhead due to the position sketch is

$$O_{\text{position}} = N_p \cdot (1 + 16) + 4 + 4 + 64.75\,\text{bytes}. \tag{8.8}$$

For the velocity, we assume an average velocity, which is added by each vehicle that contributes to the sketch. Therefore the average velocity V is added $x \cdot D$ times. Other than that, the approximation for the velocity FM sketch is analogous to the position sketch approximation:

$$N_v = \min\left(m \cdot \log \frac{xDV\rho}{m}, m \cdot w\right) \tag{8.9}$$

and

$$O_{\text{velocity}} = N_v \cdot (1 + 16) + 4 + 4 + 64.75\,\text{bytes}. \tag{8.10}$$

The total overhead of the aggregation phase is then

$$\begin{aligned} O_{\text{sketches}} &= O_{\text{count}} + O_{\text{position}} + O_{\text{velocity}} \\ &= (N_c + N_p + N_v) \cdot (1 + 16) + 4 + 4 + 64.75\,\text{bytes}. \end{aligned} \tag{8.11}$$

Note that only a single aggregate signature, reference point, and time period are required, because signatures on all sketch bits can be aggregated and time period and reference point are identical for all signatures.

During the finalization phase, the additional overhead is determined by the added multi-signature. Moreover, the finalizing vehicle's position is included to calculate the aggregate area, and the list of public keys that corresponds to the multi-signature participants. However, many public keys used in the finalization phase will overlap with keys used in the aggregation phase and need not be added again. Asymptotically, the signers of the sketch bits are a subset of the multi-signature contributors. The reason is that only some vehicles set sketch bits to 1, but all vehicles contribute to the multi-signature. Therefore, we reduce the number of extra public keys by the combined number of 1-bits in all sketches ($N_c + N_p + N_v$). The additional overhead is, therefore,

(1) position
(2) public key
(3) multi-signature

$$O_{\text{finalization}} = \underbrace{4}_{(1)} + \max(0, xD - (N_c + N_p + N_v)) \cdot \underbrace{16}_{(2)} + \underbrace{64.375\,\text{bytes}}_{(3)}. \tag{8.12}$$

In the full scheme, the total overhead is the combined overhead of the aggregation and finalization phase. Hence,

$$O_{\text{full}} = O_{\text{sketches}} + O_{\text{finalization}}. \tag{8.13}$$

In the lightweight scheme, only the finalization phase incurs security overhead. For the aggregation phase, only the total size of all sketches needs to be added. Of course, there is no overhead reduction due to overlapping public keys. Hence,

(1) position
(2) public key
(3) multi-signature

$$O_{\text{lightweight}} = 20\,\text{bytes} + \underbrace{4}_{(1)} + xD \cdot \underbrace{16}_{(2)} + \underbrace{64.375\,\text{bytes}}_{(3)}. \tag{8.14}$$

Figure 8.4 shows the security overhead of the full scheme for a low vehicle density and a high vehicle density setting and different average velocities. We assume that the sketch sizes are set as shown in Table 8.1 and that reference points are 10 km apart ($P = 5000\,\text{m}$). The average velocity is $V = 5\,\text{km/h}$ and $V = 50\,\text{km/h}$, and the vehicle density is $D = 0.04$ and $D = 0.16$ vehicles per meter. All parameters are equivalent to the settings we chose for the security evaluation in Section 8.5. The x-axis shows the geographical area covered by an aggregate, and the y-axis shows the corresponding approximate overhead consumed by security meta-data in bytes. The dotted line shows the MTU of 1500 bytes for reference.

Evidently, the scheme requires considerable overhead. The major influence on the overhead is the vehicle density D. The different average velocity V also changes the overhead, but it is negligible compared to the vehicle density. The reason is that the vehicle density influences all three sketches, whereas the velocity only influences the number of 1-bits in the velocity sketch. Moreover, the overhead in all configurations is logarithmic for geographically small aggregates and then becomes linear. During the logarithmic phase, the overhead is

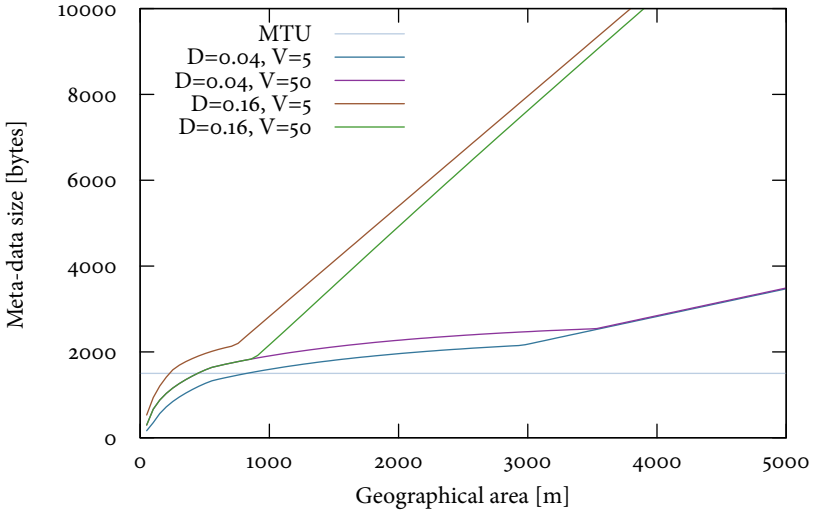


Figure 8.4: Security overhead of full scheme for different traffic scenarios.

determined by the number of 1-bits in the sketches. As shown in Equations (8.5) and (8.7), the number of 1-bits grows logarithmically. In this phase, most of the vehicles that contribute to the multi-signature also set a sketch bit to 1, so the extra overhead for public keys to verify the multi-signature is small. At some point the sketches reach their maximum capacity, that is, almost all sketch bits are set to 1. For aggregates larger than that, the major overhead factor are the additional public keys that are necessary for multi-signature verification. Note that the aggregated values will become inaccurate when the sketches reach their capacity limit. On the other hand, larger sketches incur more overhead due to the higher amount of signatures and public keys.

In all scenarios, the amount of verification meta-data quickly saturates the MTU. In the lowest density and velocity setting, at most 900 m can be covered with one packet. Therefore, the full scheme should only be used in situations where attacks are likely and a high amount of integrity protection is required. The lightweight scheme, as shown in Figure 8.5, requires less overhead. Still, at most 2200 m can be covered with a single packet. Therefore, both the full scheme and the lightweight scheme should not be used in isolation. Rather they need to adapt to different attack probabilities and used in combination with other schemes that use less overhead. Still, especially the full schemes serves to demonstrate that protection of aggregation against attacks by a variety of attackers is possible and can be integrated with duplicate filtering.

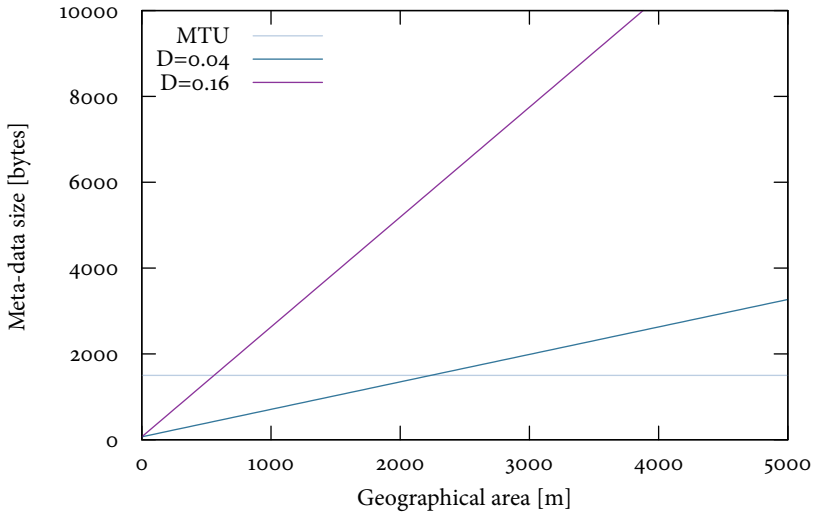


Figure 8.5: Security overhead of lightweight scheme for different traffic scenarios.

8.7 ADAPTIVITY

Fundamentally, the multi-signature scheme can be adapted in the same way as the atomic-observation-based scheme (see Section 7.7). That is, adaptation can be static, based on context, or based on aggregate contents. However, the parameters that can be adapted are different. Namely, we can adapt the following parameters to achieve different trade-offs between security and bandwidth usage.

SKETCH SIZES Larger sketches result in less hash collisions when adding elements, which results in more signed 1-bits from different vehicles. Due to the sketch construction, the number of signatures increases logarithmically with the sketch size for FM sketches and converges exponentially towards the size for LC-sketches.

MULTI-SIGNATURE PARTICIPATION In the basic implementation, all vehicles contribute to the multi-signature during the finalization phase. Alternatively, only a fraction of vehicles can participate.

LIGHTWEIGHT VS. FULL SCHEME Depending on context, the protection during the aggregation phase can be used or not.

For the LC-sketch that represents the vehicle count, an increased number of sketch bits directly results in less hash collisions. That means, more different vehicles attach signatures to the sketch and confidence in the sketch correctness increases. Each vehicle adds at most one 1-bit, so attackers that alter the sketch by more than one bit can always be detected.

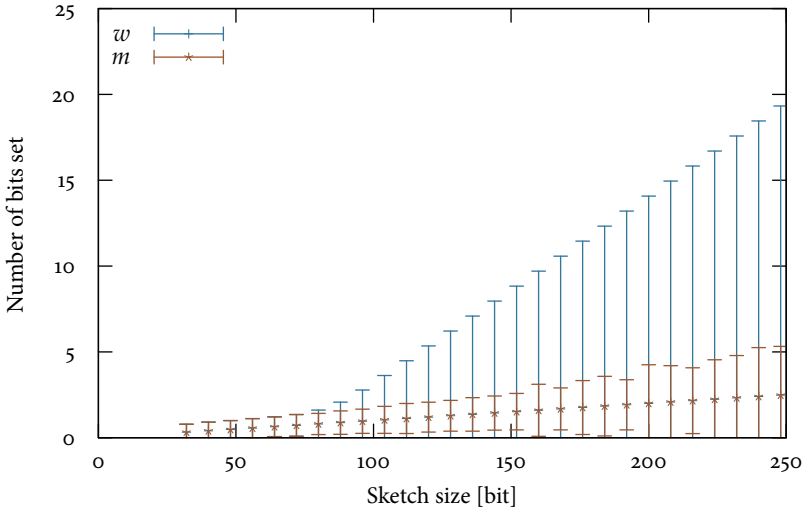


Figure 8.6: Average number of 1-bits set per sketch participant.

For the FM sketches, the situation is different. Both FM sketches represent sums. Therefore, each vehicle adds more than one 1-bit. Attackers can exploit this to influence the sketch outcome. When the FM sketch size is adapted, more bits are available for signing and hash collisions are less likely. But because each vehicle adds a sum to the sketch, the higher number of available bits also makes it more likely that more 1-bits are set by the same vehicle. Figure 8.6 shows this effect. We simulated 100 vehicles that each add a value of 50 km/h to the sketch. We implement an algorithm that maximizes the diversity of signatures on the sketch: when a vehicle wants to set a bit to 1 that was already set to 1 earlier, it adds its signature to the bit only if the new vehicle has signed less bits than the old vehicle. The varying total sketch sizes are achieved by altering the size of a single bit array (w) or altering the number of bit arrays per sketch (m). The two data sets show varying numbers for either w or m; the respective other value is set to 8. The y-axis shows the average number of bits set to 1 per vehicle. While the average number of bits set is the same for variations of w and m, the standard deviation is more predictable when adapting m. We therefore argue that adapting m is better for integrity protection. Due to the low standard deviation, a data-consistency mechanism can check whether each contributing vehicle has set the same number of bits to 1.

Figure 8.7 shows an additional benefit of adapting m over adapting w. Here, the y-axis shows the number of unique contributors, that is, the number of unique vehicles that added a signature to the sketch. When adapting w, the number of unique contributors stagnates at 65, which is due to the geometric hash function that is used to add values to a bit array in FM sketches. When adapting m, however, the number of contributors increases with the sketch

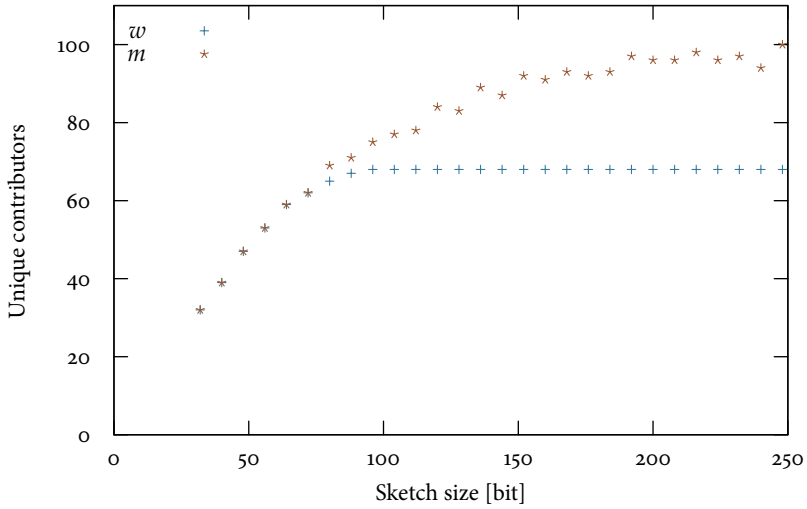


Figure 8.7: Number of unique sketch participants.

size. The reason is that the bit array that an element is added to is chosen uniformly from the number of bit arrays m. Therefore, the chance that a vehicle contributes is higher than when adapting w.

Hence, it is more useful from an integrity protection viewpoint to adapt m instead of w. However, m cannot be adapted during the evolution of aggregates. When an aggregate is created with a specific value for m, it cannot be changed during the aggregate's lifetime. For adapting m, vehicles therefore need a common understanding of the currently required value for m. Contrarily, w can both be increased and decreased during an aggregate's lifetime. These alterations are enabled by the geometric hash function. The w-th bit of a sketch with size w is only set to 1 with probability 2^{-w}; the $(w+1)$-th bit will be set with probability $2^{-(w+1)}$. Due to this lowering probability for increasing w, the chance that the sketch approximation is altered by adding more bits is considerably lower than for a uniformly distributed hash function. Likewise, bits can be removed from an existing sketch without altering its approximation as long as the initial uninterrupted sequence of 1-bits is kept intact. In summary, m-adaptation offers better integrity protection, but w-adaptation offers more flexibility for adaptations during aggregate lifetime.

Adapting the number of multi-signature participants is similar to adapting S in the atomic-value scheme (Chapter 7). The number of signers directly influences the security overhead due to the changed number of public keys required for signature verification. When only a fraction of all possible vehicles contribute to the multi-signature, the overhead reduction is a constant factor, and the overhead size is still linear in the aggregate's geographical area. In addition to using a fraction of signers, we can also define a target number of vehi-

cles that need to participate in the multi-signature and stop signing after that threshold is reached. This second option was not possible for the atomic-value scheme, because there, the atomic witnesses need to be uniformly distributed over the aggregate's geographical area. Now, we can exploit the opposing dissemination directions during the aggregation and finalization phase. When an aggregate enters the finalization phase, it must already represent the whole aggregate area. Therefore, a fixed small number of multi-signers suffices to ensure aggregate integrity. An attacker further upstream cannot modify the aggregate, because the existing multi-signature is bound to the whole aggregate content, whereas in the previous scheme, each signature attests to a single atomic value.

Similarly, because of the distinct dissemination phases, a possible adaptivity is the choice between using the full scheme or the lightweight scheme. If the lightweight scheme is used, the aggregation phase is unprotected. Therefore, vehicles can only use their local sensor readings as plausibility checks during the finalization phase. Consequently, multi-signature participants should be uniformly distributed throughout the aggregate's geographical area to ensure that the claimed velocity is valid for the whole claimed area. In this constellation, a fixed threshold of signers does not suffice to protect the aggregate, but a fraction of signers can still be used to adapt overhead.

In summary, the multi-signature-based scheme offers more options for adaptivity, but configuration is more complex and some parameters' influence on integrity protection is not intuitive. However, the scheme's security can be scaled to withstand strong attackers. We will discuss different adaptation strategies and their implementation in Chapter 11.

8.8 SUMMARY

We have presented a scheme that exploits predominant information dissemination directions to achieve integrity protection. Observing different aggregate lifetime phases, we implemented an agreement phase that is separate from aggregate finalization and further dissemination. During the agreement phase, we use LC-sketches and FM sketches to implement duplicate insensitivity, which achieves better information quality. Moreover, the integrity protection is better than that of the scheme presented in Chapter 7, because more vehicles can contribute to a multi-signature that attests to an aggregate's correctness. Moreover, the distinct and opposing dissemination directions during aggregation and finalization phase allow for a wider range of data consistency checks. Finally, the scheme offers several parameters for adaptivity, which can be configured to implement different trade-offs between security overhead and geographical area that can be covered. On the downside, the scheme's overhead is considerably higher than that of the previous scheme when full protection is implemented.

9

9.1 OVERVIEW

The clustering mechanism description (Section 9.2) is a revised version of our publication [4]. Preliminary investigations for the clustering mechanism were conducted as part of a supervised master thesis [226]. The security mechanism (Sections 9.4 and 9.5) is a revised and extended version of our publication [9].

In the following scheme, we eliminate two drawbacks of the scheme we discussed in Chapter 8. The first drawback is that information is forwarded through the aggregate region twice to enable average velocity agreement. The double forwarding is necessary due to the fully flexible information dissemination, but it introduces additional overhead. Second, information is attested using multi-signatures. While identity based signatures can be used to limit the multi-signature overhead, the required space still grows linearly in the number of aggregate participants.

To achieve better efficiency, we introduce an additional dissemination structure using a clustering mechanism. Clustering has received some attention in the VANET research domain [77, 143], but it is often argued that cluster structures cannot be maintained due to high vehicle mobility. As the basis for the cluster-based security mechanism, we design a clustering mechanism that is specifically tailored to aggregation. Vehicles are clustered according to their relative velocity to find stretches of similar speed. Because velocity is used for clustering, the scheme is mainly applicable to aggregation of traffic information. Other forms of information, such as parking spot availability, may be benefit from our security protocol, but would not benefit from the homogeneous clusters.

The clustering mechanism enables agreement on average velocities within the cluster structures. We assume that clusters form a trusted unit in the sense that it is infeasible for an attacker to control the majority of vehicles within one cluster. This trust assumption can be problematic in smaller clusters. Therefore, we only consider clusters with a minimum size of τ as trustworthy in the cluster-based security protocol. Because clusters with a minimum size of τ are trustworthy, it is sufficient to create a proof protocol between two neighboring clusters. Once a neighbor cluster – which is again a trusted unit – has verified an aggregate, the proof information of the original cluster can be stripped.

In the following, we first present and evaluate the underlying clustering mechanism in Sections 9.2 and 9.3. We then explain our security approach for intra-cluster agreement and inter-cluster dissemination in Section 9.4. We evaluate our approach regarding the achieved security and required bandwidth overhead in Sections 9.5 and 9.6, respectively. In Section 9.7, we discuss how our mechanism can be adapted to different traffic situations.

9.2 SIMILARITY-BASED CLUSTERING

In general, the goal of a VANET clustering protocol is to identify groups of vehicles that share certain properties, such as geographic location or velocity.

Once groups are formed, usually one vehicle per group assumes a special role, the so-called cluster head. The cluster head performs maintenance operations, such as allowing new vehicles to join the cluster or handling vehicles that leave the cluster because their properties become too different from the rest of the group. Often, clustering protocols use geographic location to form clusters. But heterogeneous velocities may lead to unstable clusters if geographic positions are used as cluster formation criterion.

We propose a clustering protocol design that results in more stable clusters in vehicular networks by integrating mechanisms from current in-network aggregation schemes. In Section 3.6.3, we argued that an in-network aggregation scheme generically consists of a decision, fusion, and dissemination component. In particular, the mechanisms often used for aggregation decision components for traffic information systems can be applied to clustering protocols to find more stable clusters. Current aggregation mechanisms [e. g., 51, 3] use flexible decisions based on relative velocity and other parameters in addition to vehicle location. By doing so, they improve on earlier approaches for in-network aggregation, which were mostly based on location intervals [e. g., 170]. Therefore, we expect velocity-based clustering to improve cluster stability compared to clustering protocols that are based on location intervals. Once stable clusters can be assumed, clustering can improve the fusion and dissemination components of in-network aggregation. When cluster members stay within communication range for some period of time, they can agree on their average velocity and other values, and the cluster head can eliminate duplicates to improve information quality, as discussed in Section 3.4.3. Cluster heads can also organize information dissemination to neighbor clusters, which lowers the bandwidth overhead that opportunistic dissemination mechanisms typically incur. Finally, clustering is beneficial for integrity protection, because cluster members can interactively agree on and create a proof of aggregated values.

Integrating aggregation decisions and clustering.

The key factor for integration of clustering and in-network aggregation is the adaptation of aggregation decisions to the cluster formation process. Generally speaking, our goal is to cluster vehicles that share similar characteristics. For instance, vehicles within a traffic jam or vehicles driving in a convoy will result in very stable cluster structures, because they share similar locations and mobility patterns. We argue that dynamic characteristics of nodes, such as their relative velocity and driving patterns, should be given a higher priority for clustering decisions than their geographical location parameters. The reason is that heterogeneous vehicle velocities are one of the main reasons why existing work often argues that clustering fails in dynamic networks [110]. In this context, we use the *relative average velocity* (v_{rel}) between nodes as the main clustering metric. In addition, we also use the transmission range as secondary clustering criterion. That is, clusters are established exclusively between nodes located in each other's proximity and within single-hop transmission range to the cluster head. According to our hypothesis, nodes with similar velocities $v_i \approx v_j$ (i. e., $v_{rel} \longrightarrow 0$) will create more stable clusters.

9.2.1 *Cluster Formation*

We base our message exchange and cluster formation protocol on cluster-based location routing (CBLR) [151] but replace CBLR's cluster joining criterion to implement velocity-based clustering. Cluster formation starts with each node entering the so-called cluster undecided (CU) state. While in the CU state, nodes broadcast a HELLO message to announce their presence to possible neighbors and start a timer T_{rand}, within which they waits for replies, including a HELLO message from a possible adjacent cluster head (CH). Note that HELLO messages can be piggy-backed on existing beacons (e. g., CAMs [195]) to minimize clustering overhead. If during this time interval a CU node receives a HELLO packet from a CH, it automatically becomes a cluster member (CM) of that CH's cluster. In case a node receives two HELLO packets, it joins the cluster with the more similar mobility pattern. If the node receives no messages, it becomes a CH.

Velocity Threshold Condition.

In order to implement relative-velocity-based clustering, we introduce a *velocity threshold condition*. Intuitively, whenever a CU node is about to connect to a CH, the two exchange their average velocities using HELLO packets. Both nodes then compare the local and received values and check if the difference is less than or equal to the preset threshold. The joining node's average velocity is compared with the average velocity of *all* nodes already participating in the cluster ($v_{avg(cluster)}$) to increase cluster stability. Only if the average velocity of a CU node and the cluster are similar, the *velocity threshold condition* is met and the node joins the cluster. Namely, a CU joins if:

$$v_{rel} = |v_{avg(node)} - v_{avg(cluster)}| \leq v_{threshold}. \tag{9.1}$$

Here, we require that velocities are represented as one-dimensional values. In case of multiple nearby highways, the criterion can be extended by considering a highway id as additional criterion. In city scenarios, velocities can be expressed as vectors to distinguish streets in different directions. Both extensions can be added without changing the mechanism's concepts.

To further increase cluster stability, we use an exponential moving average of past speed values for clustering decisions, which is based on the last n measurements, instead of only the node's current speed. This prevents vehicles that only shortly drive the with the same speed as the existing cluster, which is a drawback of Rawshdeh and Mahmud's proposal [143]. Consider as example a vehicle that accelerates to overtakes another vehicle but generally drives slower than the cluster. Also, a weighting factor α is used to give a higher importance to the newest velocity value ($v_{current}$). Namely,

$$v_{avg_k} = \alpha \cdot v_{current} + (1 - \alpha) \cdot v_{avg_{k-1}}, \tag{9.2}$$

where $v_{avg_{k-1}}$ is the last calculated node average velocity and v_{avg_k} is the currently calculated average velocity.

9.2.2 *Cluster Maintenance and Re-clustering*

Once nodes have joined a cluster, the *velocity threshold condition* is constantly re-verified at each periodic HELLO packet exchange between the CH and its CMs. If the difference does not exceed the threshold defined in Equation (9.1), the CM stays in the cluster. However, if this condition is not met, the CM will leave the cluster. In case the CH itself changes velocity, it performs the same checks. However, the CH cannot simply leave the cluster. Instead it determines the CM with the velocity that is closest to the cluster average $v_{avg(cluster)}$ as soon as the CH violates the threshold condition. The CH then broadcasts a packet, which hands over CH duty to the CM with the closest velocity. The old CH then enters the CU state until it encounters a new cluster or neighbor vehicle.

Merging clusters.

Whenever two CHs enter each other's communication range, they consider whether to merge their clusters. The CHs compare their cluster average velocities and check whether the *velocity threshold condition* is satisfied:

$$|v_{avg(cluster\_1)} - v_{avg(cluster\_2)}| \leq v_{thresh}. \tag{9.3}$$

Only if the threshold condition is fulfilled, will the clusters initiate the merging procedure.

In order to decide which CH of the previous clusters will be the cluster head of the merged cluster, both CHs compare their *node* average velocity with the other *cluster's* average velocity:

$$\delta_1 = |v_{avg(CH1\_node)} - v_{avg(cluster\_2)}|, \tag{9.4}$$

$$\delta_2 = |v_{avg(CH2\_node)} - v_{avg(cluster\_1)}|. \tag{9.5}$$

If $\delta_1 < \delta_2$, CH1 becomes the new cluster head; otherwise, CH2 becomes the new cluster head. We use the cluster average velocity in this situation, because it characterizes the dynamic mobility properties of the cluster as a whole. The CH election is won by the CH whose node average velocity is closer to the cluster average velocity of the other cluster. This selection process ensures that the new CH will move along with the existing cluster for a longer period of time.

9.3 CLUSTER STABILITY EVALUATION

In the following, we evaluate our clustering mechanism by comparing the expected cluster stability of our scheme and a location-based clustering scheme, which clusters purely based on location of vehicles. The protocols' main difference is their cluster joining criteria:

- The baseline protocol, *location-based clustering*, clusters *all* vehicles that move into mutual communication range, independently of their velocity.
- Our protocol, *velocity-based clustering*, clusters *only* vehicles that satisfy the velocity threshold condition, i. e., that have similar velocities, as explained in Section 9.2.

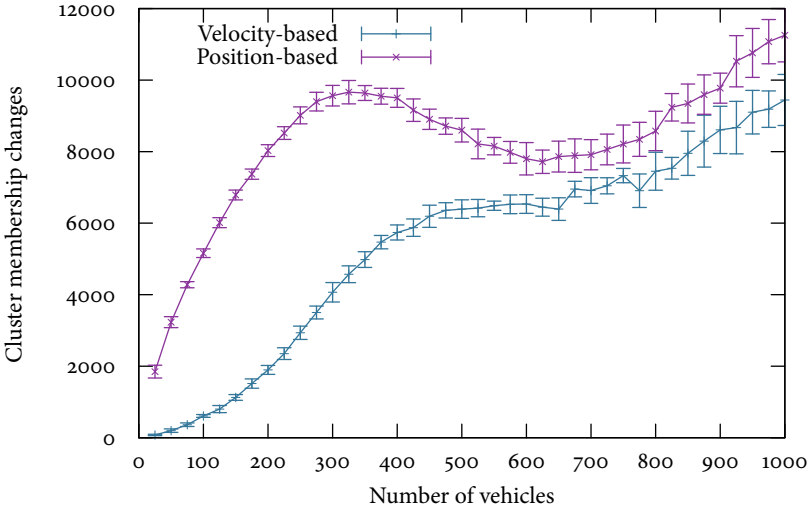


Figure 9.1: Average cluster membership changes for position-based clustering and velocity-based clustering.

Our main reason for introducing a clustering protocol is to provide stable structures for the security mechanism we present in Section 9.4. Therefore, we focus our evaluation on determining cluster stability. For a quantitative comparison of both clustering approaches, we use cluster membership changes. Whenever a vehicle joins a cluster or leaves a cluster in the simulation, the membership changes counter is increased by 1.

Figure 9.1 shows the cumulative number of membership changes for both clustering mechanisms on a 10 km stretch of highway with 3 lanes. Vehicles are initially distributed randomly along the highway, but they approach a road block at the 10 km mark, which leads to a traffic jam towards the end of the simulation runs. The x-axis shows the vehicle density expressed as total number of vehicles in the simulation, and the y-axis shows the corresponding total number of membership changes per simulation run with standard deviation, averaged over 10 simulation runs.

Velocity-based clustering performs better than position-based clustering.

Velocity-based clustering consistently performs better than position-based clustering. Moreover, the difference is bigger for lower vehicle densities. The reason is that higher vehicle densities lead to slower and more homogeneous traffic along the highway. When vehicles move slower, they do not change positions as rapidly, which is beneficial for position-based clustering. But velocity-based clustering also benefits from more homogeneous traffic. Our results show that for 300 or more vehicles, the cluster membership changes decreases despite increasing numbers of vehicles. Similarly, the number of membership changes does not change for position-based clustering for densities between 500 and 600 vehicles.

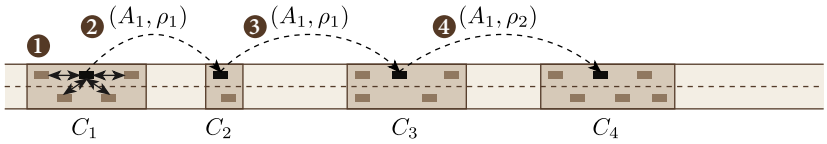


Figure 9.2: Overview of clustering security components. Cluster C_1 creates a proof ρ_1, which C_2 forwards unmodified, because C_2 is too small. However, C_3, a larger cluster, creates a new proof ρ_2 and discards ρ_1.

For higher vehicle densities, that is, above 600 vehicles, both clustering mechanisms show linearly increasing membership changes. Velocity-based clustering still outperforms position-based clustering, and their difference remains constant. The reason for this behavior is the saturated communication channel for high vehicle densities. In both implementations, vehicles periodically exchange HELLO messages with their cluster heads. Vehicles will leave their cluster if they did not receive a HELLO message from their CH after 3 beacon intervals. As more messages get lost due to MAC contention, cluster membership changes increase despite vehicles not physically moving. Simulation results underline that velocity-based clustering leads to at least as stable clusters as position-based clustering, independent of vehicle density. For many vehicle densities, velocity-based clustering leads to significantly fewer cluster membership changes.

9.4 SECURITY APPROACH

Our cluster-based security mechanism is built on the assumption that clusters of vehicles serve as a trustworthy unit. That is, we assume that it is unlikely that attackers compromise the majority of vehicles within a cluster. In Section 4.4, we discussed that an attacker in our model can compromise the key material of at least one vehicle. The total number of controlled vehicles, however, will be small, because attackers are assumed to require full physical access to compromise vehicles and their key material. Let n be the number of vehicles that an attacker can compromise. Then clusters with at least $\tau > 2n$ members can be assumed trustworthy if majority decisions are used to agree on a cluster-wide aggregated view.

The goal of our integrity protection mechanism is to make the majority decision verifiable within the cluster and to create a bandwidth-efficient proof of the decision to be disseminated to neighboring clusters. Moreover, the mechanism needs to provide a fallback option for clusters with fewer than τ members. Figure 9.2 shows an overview of the mechanism. Four clusters $C_1, \ldots C_4$ are distributed along a road. Vehicles in C_1 created an aggregate A_1, which is disseminated to C_2 and then further downstream. At the same time, other clusters may create aggregates that may be merged with A_1; we omit these in the figure for clarity. The security mechanism is comprised of four main components.

1. Within C_1, vehicles communicate their atomic observations to the cluster head which calculates an aggregated value A_1 and disseminates it within the cluster. All cluster members can verify the aggregated value based on atomic observations from other cluster members that they overheard by applying models for plausible traffic behavior (cf. Section 4.6.3).

2. C_1's members then interactively create a proof ρ_1 that can be disseminated to neighboring vehicles along with A_1. The semantic of ρ_1 is that it proves that at least $n+1$ vehicles have participated in a majority decision and are confident that A_1 is a correct representation of the traffic situation on the road segment covered by C_1.

3. C_2 cannot be assumed trustworthy, because its size is smaller than τ. Therefore, C_2 can only participate in forwarding information from C_1 to C_3. It does not create an own proof but forwards (A_1, ρ_1).

4. C_3 is a larger cluster (i. e., has at leatst τ members) and is, therefore, assumed trustworthy. Hence, C_3 will not forward ρ_1 but instead create an own proof ρ_3 that replaces ρ_1. The semantic of ρ_3 is that the (honest) majority of C_3's members have verified ρ_1's correctness. In addition C_3 may have merged A_1 with its own aggregate A_3 in case they are both part of the same traffic situation.

The components can be grouped in the interaction *within* the cluster between cluster head and cluster members, which we detail in Section 9.4.1, and the dissemination protocol *between* clusters, which we discuss in Section 9.4.2.

9.4.1 *Intra-cluster agreement*

Within a cluster, our security mechanism is based on data consistency mechanisms. Each cluster member i periodically broadcasts its current velocity (v_i), location (p_i), time (t_i), and a signature on the values, as well as its public key and certificate:

$$o_i = ((p_i, t_i), v_i, \sigma_i(\cdot), pk_i, cert_i). \tag{9.6}$$

The cluster head collects all observations from all n cluster members and calculates the cluster's aggregate:

$$A := (([\min_{1\le i\le n} p_i, \max_{1\le i\le n} p_i], \max_{1\le i\le n} t_i), \frac{1}{n}\sum_{i=1}^{n} v_i). \tag{9.7}$$

The cluster aggregate is then broadcasted by the cluster head in subsequent beacons. Each cluster member verifies whether the claimed average cluster speed deviates from the own velocity by at most the cluster joining threshold $v_{threshold}$. Also, each member verifies that the claimed aggregate time is current and that the own location is within the interval claimed by the aggregate. Moreover, the

aggregate interval may not be significantly larger than $2r$ where r is the approximate wireless range, because clusters are only formed between vehicles within communication range of the cluster head. If all of these checks hold, the cluster member signs the cluster aggregate and attaches $\sigma_i(A)$ to subsequent beacons.

The cluster head collects all signatures $\sigma_i(A)$. If at least τ signatures can be collected, we can assume that A is a correct representation of the situation in the cluster. While the cluster head is responsible for collecting all signatures $\sigma_i(A)$, all other cluster members can overhear the signed messages, as well. We exploit this overhearing to detect malicious cluster heads. If a cluster head disseminates an aggregate A, and cluster members do not overhear at least τ signatures within a certain time period, they will ignore all further communication from the cluster head. Moreover, all honest cluster members change their state to CU and re-start the clustering process, excluding the malicious cluster head.

Once the cluster head has collected at least τ signatures on the aggregate A, it can communicate the aggregate content to neighbor clusters to further disseminate the aggregated values and possibly merge them with other aggregates, implementing hierarchical aggregation.

9.4.2 *Inter-cluster dissemination*

Proof of τ to neighbor clusters.

When a cluster aggregate A is disseminated to neighbor clusters, they cannot use data consistency alone to judge the correctness of A's values. Instead, we create a proof that *at least* τ vehicles within the cluster that created A checked its consistency and disseminate that to the neighbor cluster. Ideally, the proof should represent *all* $n \geq \tau$ members of the original cluster.

The easiest way to create such a proof would be to attach all atomic signatures $\sigma_i(A)$ to the aggregate and disseminate them to the neighbor cluster. In traffic jams and other homogeneous traffic situations, however, hundreds of vehicles may form a cluster. Therefore, the naïve approach would consume too much bandwidth. Instead, we use an estimator to prove the number of cluster members agreeing to A. The technique to use estimators in this way was originally proposed by Hsiao et al. [82] to validate event reports in VANETs. We extend their approach to the more dynamic setting of in-network aggregation, as well as to support hierarchical aggregation.

See Section 5.3 for a discussion of Hsiao et al.'s mechanism.

Estimators are used in the database domain to approximately count distinct elements within a stream in a single pass and with a sub-linear memory complexity. Hsiao et al. [82] use the z-smallest estimator to create a proof of an event. We instead use the newer HyperLogLog data structure proposed by Flajolet et al. [69] as an extension of FM sketches and further developed by Google [81]. In addition to providing a bandwidth-efficient proof of participant numbers, the estimator provides our mechanism with a means to detect duplicates when aggregates are merged hierarchically.

In applying estimators to integrity protection, the goal is to select a subset of size $s \ll n$ of all cluster member signatures to be disseminated to the neighbor

cluster such that the probability that an attacker can successfully forge s signatures and still convey n total cluster members is negligible. To achieve that, we exploit that a hash function that cannot be determined by the attacker is used to create the estimator data structure. And we use the hash function and the estimator data structure to select the subset s of signatures that are disseminated to the neighbor cluster.

Namely, we create an integrity-protected HyperLogLog estimator of the cluster size as follows. The HyperLogLog data structure consists of $m = 2^b, b \in \mathbb{Z}_{>0}$ registers. Each register $M[i], 1 \leq i \leq m$ is initially set to 0. To add a value v, a single 32-bit hash function is used

$$h : \mathcal{D} \to \{0,1\}^{32}. \tag{9.8}$$

Then, the first b bits of $h(v)$ are used to determine the register $M[i]$ to use. Of the remaining bits, the length l of the initial uninterrupted sequence of 0 bits is counted. The register $M[i]$ is then updated as follows:

$$M[i] = \max(M[i], l). \tag{9.9}$$

Note that the estimated probability of l initial uninterrupted zeros occurring is 2^{-l}. Thus, the number of distinct elements per register $M[i]$ can be estimated as 2^l. The estimated total number of distinct values is then calculated as the harmonic mean of each register's value:

$$E := \alpha_m m^2 \left(\sum_{j=1}^{m} 2^{-M[j]} \right)^{-1}, \tag{9.10}$$

where α_m is a correction factor that depends on the number of registers m. The expected error of the estimator is in the order of $1.04/\sqrt{m}$ [69].

To build the HyperLogLog estimator for the number of cluster participants, we add the tuple

$$(A, pk_i) \tag{9.11}$$

for each cluster member i to the estimator. Let $\{\sigma_{\pi(1)}(A), \ldots, \sigma_{\pi(m)}(A)\}$ be the m signatures for which adding their corresponding tuple contributed the maximum register values. Then

$$\rho = (\sigma_{\pi(1)}(A), \ldots, \sigma_{\pi(m)}(A)) \tag{9.12}$$

is the proof that is communicated to the neighbor cluster. Note that the proof size is at most m signatures, independent of the cluster size. The receiving cluster verifies that all m signatures are valid and recreates the estimator data structure using the public keys and A. It then calculates the estimate E and verifies that $E \geq \tau$. Because the HyperLogLog data structure is determined by adding the aggregate value A and the public keys pk_i corresponding to a correct signature on A, an attacker cannot inflate A. The reason is that A is determined

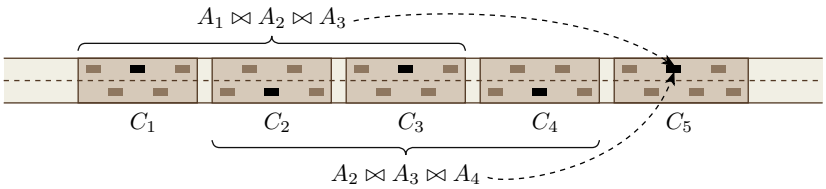


Figure 9.3: An example of overlapping aggregates received by a cluster.

by the information that the attacker wants to disseminate. That is, the attacker cannot choose A at will without altering the traffic situation A represents. Even if the attacker would be able to choose public keys at will, they would still need to brute force the hash function to find public keys that result in desired hashes with long runs of 0 bits [82].

Hierarchical aggregation with duplicate detection.

So far, we discussed the exchange of aggregates between directly adjacent clusters. However, the goal of aggregation is to disseminate information farther than that. As shown in Figure 9.2, other clusters will disseminate not only their own aggregates but also a subset of aggregates received from other clusters. Moreover, clusters may merge their own aggregates with aggregates from other clusters and only forward the merged result. Such merging will happen if multiple clusters are part of the same traffic situation, such as a long traffic jam.

For example, consider the situation shown in Figure 9.3 where 5 clusters are all part of the same traffic jam. Here, clusters C_1, C_2, and C_3 have merged their aggregates, as have C_2, C_3, and C_4. Cluster C_5 receives both $A_x = (A_1 \bowtie A_2 \bowtie A_3)$ and $A_y = (A_2 \bowtie A_3 \bowtie A_4)$. If C_5 simply combines both merged aggregates, the values of A_2 and A_3 would be counted twice, resulting in a biased aggregate, as discussed in Section 3.4.3. To prevent such biases, we again exploit the HyperLogLog data structure. Using estimators to detect duplicates and eliminate bias has been applied in earlier work, such as Lochert et al. [114], as well as our mechanism discussed in Chapter 8. Here, we use the *distinct* counting capability of estimators to estimate the number of elements in the intersection of two aggregates.

Whenever two aggregates A_1 and A_2 are merged, the estimator for the number of participants is updated as follows. Let M_1 be the estimator for A_1's participants and M_2 be the estimator for A_2's participants. Then

$$\forall 1 \leq i \leq m : M[i] = \max(M_1[i], M_2[i]). \tag{9.13}$$

As discussed by Flajolet et al. [69], this operation is equivalent to counting the distinct elements in the merged lists in the first place. Let $E(A_x \bowtie A_y)$ be the estimate of two aggregates' union, calculated as explained above.

Suppose a cluster now receives the two aggregates A_x and A_y, which are the result of merging several cluster aggregates. The number of vehicles in their intersection can be estimated by

$$I = E(A_x) + E(A_y) - E(A_x \bowtie A_y). \tag{9.14}$$

If $I > 0$, it is likely that merging A_x and A_y would introduce a bias in the merged aggregate. Therefore, we choose to keep these aggregates separate and disseminate them separately to neighboring clusters.

Proof of hierarchically aggregated values.

When values are aggregated hierarchically, the original proof information becomes invalid. The participants of the first clusters signed their corresponding maximum entries, which are changed due to the hierarchical aggregation. Rather than combining the proofs, we again exploit that a cluster with at least τ members is considered trustworthy to create a new proof. Suppose a cluster C_2 has merged aggregates A_1 and A_2 as explained above and wants to communicate $A_{1,2} = (A_1 \bowtie A_2)$ to a third cluster C_3. First, C_2's cluster head disseminates the merged aggregate $A_{1,2}$, as well as A_1 and A_2 to its cluster members. All cluster members verify that $A_{1,2}$ is a correct aggregation of A_1 and A_2. They also verify that A_2 corresponds to the traffic situation in the cluster. Finally, the cluster members verify that A_1 was received before and that it was accompanied by a proof ρ_1. Once cluster member i successfully performed these checks, it creates a signature $\sigma_i(A_{1,2})$ and broadcasts it.

The cluster head creates an estimator M using the received signatures, which attests to the number of cluster members in C_2. In addition, it creates an estimator M' by merging the estimators of A_1 and A_2 as explained above. The cluster head then forwards a proof ρ_2, as well as a difference-encoded estimator E' for $A_{1,2}$ to C_3, where

$$\rho_2 = (\sigma_{\pi(1)}(A), \ldots, \sigma_{\pi(m)}(A)), \tag{9.15}$$

$$E' = (M[1] - M'[1], \ldots, M[m] - M'[m]). \tag{9.16}$$

Note that C_2 only proves the number of its own cluster members, but it additionally creates an estimator for the merged aggregate. We use the proof for verification of trustworthiness, and we use the combined estimator to detect overlaps during further hierarchical aggregation, as explained above. We only use C_2's proof, because we assume that the cluster as a unit is trustworthy if it has at least τ members. Therefore, we assume that C_2's members can attest that they checked the previous cluster's proof. Only if C_2 does not merge the received aggregate will it keep the old proof information.

Dealing with small clusters.

So far, we assumed that clusters contain at least τ members and can, therefore, create valid proofs for aggregated values. If a cluster is smaller than τ members, it cannot participate in the aggregation process. That is, it can create an aggregate pertaining to the traffic situation within the own cluster and can create an estimator and a proof for it. But since less than τ members can be proven by the estimator, receiving clusters should not have high confidence in the received information. Moreover, clusters with less than τ members cannot merge aggregates at all. Still, information from small clusters may be used if can be verified by other means, such as data consistency checks. Instead, they forward the received aggregates unmodified and including the original proofs from sending clusters. This approach may incur additional bandwidth overhead in favor of higher security. However, we argue that in accordance with the attacker model (Section 4.4), τ need not be large. Because attackers need to physically compro-

mise cars to obtain key material, and since key certificates will have a limited lifetime, $\tau = 10$ may already offer sufficient protection.

9.5 SECURITY EVALUATION

Our security mechanism is built on the assumption that clusters of a minimum size τ form a trustworthy unit due to the assumption of an honest majority. Once the minimum size τ is reached, a majority vote assures correct aggregation within a cluster, as discussed in Section 9.4.1, and an integrity protected HyperLogLog sketch assures the reaching of the threshold τ to other clusters, as discussed in Section 9.4.2. The security primitives used are based on well known cryptography, such as ECDSA cryptographic signatures, which we assume to be secure against attacks. Rather than verifying the underlying cryptographic primitives, we therefore focus our security analysis on the clustering behavior of the vehicles, as well as the behavior of the sketch integrity protection.

In particular, we analyze how likely the cluster size threshold τ will be reached in different scenarios in Section 9.5.1. Based on these results, we discuss how many vehicles an attacker needs to compromise to successfully alter aggregates of typical cluster sizes, thereby implementing a `FAKE` or `CONCEAL` attack (cf. Section 4.4.3), in Section 9.5.2. Concluding the security evaluation, we discuss implications of the HyperLogLog sketch for hierarchical aggregation in Section 9.5.3.

9.5.1 *Average cluster size*

We assume that clusters form units of trust if they have a certain minimum size τ. If many clusters on a road have less than τ members, the security mechanism will have a considerably higher bandwidth overhead, because those clusters do not participate in hierarchical aggregation. To create stable and large clusters with high probability, we designed a velocity-based clustering mechanism, as outlined in Section 9.2. Figure 9.4 shows the average cluster size for different vehicle densities on a 5 km highway stretch with 3 lanes. The x-axis shows the total number of vehicles in the simulation and the y-axis shows the corresponding average cluster size with standard deviation for 10 simulation runs with randomized vehicle positioning.

Most clusters are sufficiently large.

As expected, the average cluster size increases as the number of vehicles increases. In particular, the average cluster size is at least 10 vehicles for 100 or more vehicles on the 5 km highway. That is, if there is at least one vehicle every 150 m per lane, the average cluster size will be, on average, 10 vehicles or more. Because we assume that an attacker needs to physically compromise a vehicle to obtain key material, a threshold cluster size of $\tau = 10$ may already offer sufficient integrity protection, as argued in Section 9.4. Therefore, simulation results show that the security mechanism applied to velocity-based clustering will reach the required cluster size threshold in most traffic densities. In partic-

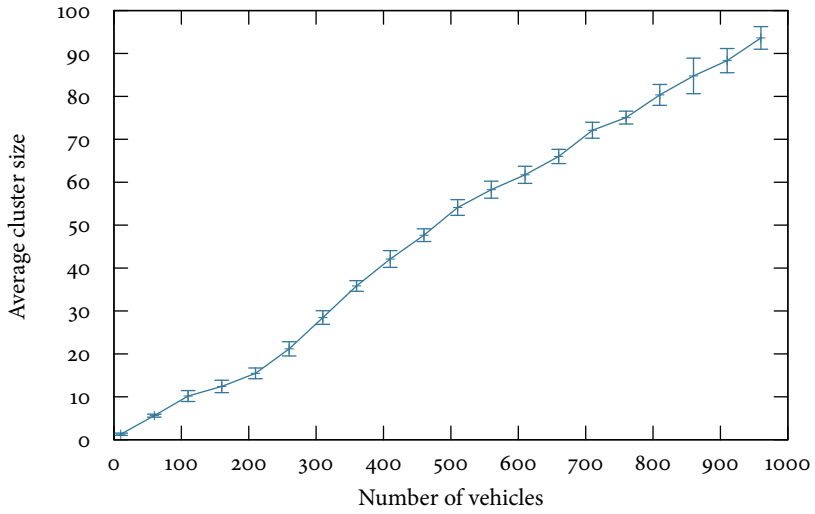


Figure 9.4: Average cluster sizes for different vehicle densities.

ular, the mechanism can benefit from large cluster sizes in high traffic density settings.

9.5.2 *Sketch manipulation*

We use a HyperLogLog sketch to represent a proof of cluster size using significantly fewer total cryptographic signatures than there are cluster members. The number of signatures used corresponds to the parameter m of the sketch. The smallest possible value is $m = 4$ [81], corresponding to $2^m = 16$ total signatures. Because the subset of included signatures is determined by a hash of vehicle public keys, the attacker cannot freely add signatures to the sketch. But if the attacker were to possess all public keys that are added to the sketch, it could create a false proof of an arbitrary cluster size.

In Figure 9.5, we analyze how likely it is that an attacker can create a false proof of cluster size. We use a HyperLogLog data structure with $2^m = 16$ registers. For each register, the signature of the public key that contributed the maximum register value is kept, as discussed in Section 9.4.2. We randomly select a subset of all vehicles to be controlled by the attacker and plot the corresponding attack success probability. An attacker is considered successful if all signatures attached to the HyperLogLog data structure are created by vehicles under attacker control. Figure 9.5 shows attack success percentage for 10 000 random clusters with different sizes. The graphs show that the success probability is sub-linear in all cases. That is, if an attacker controls n percent of all vehicles within a cluster, the success probability is much smaller than n. The at-

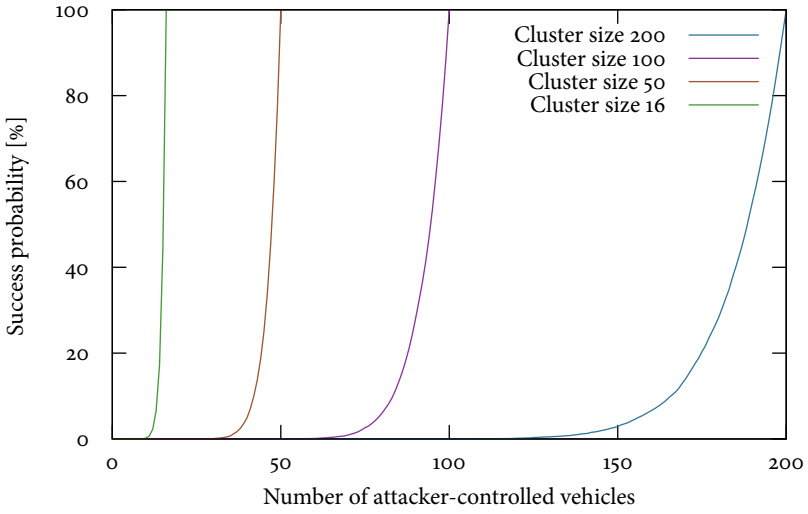


Figure 9.5: Probability of attacker success for different cluster sizes and numbers of attackers.

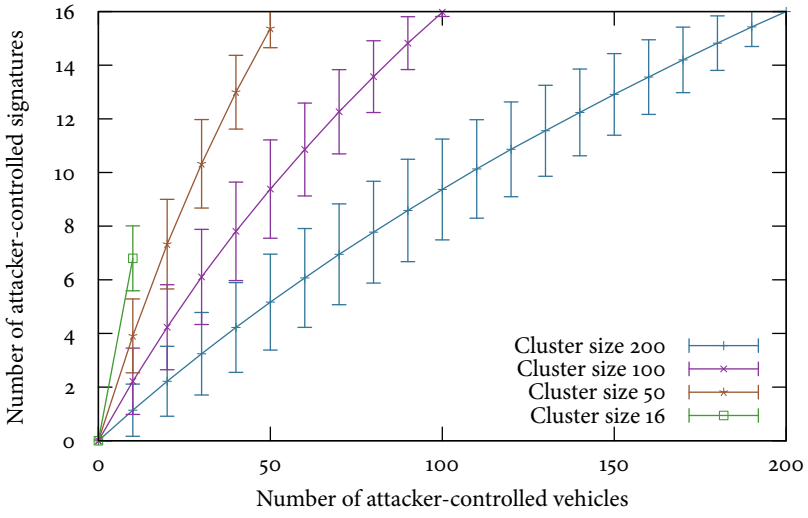


Figure 9.6: Number of attacker-controlled signatures in HyperLogLog sketch for different cluster sizes and numbers of attackers.

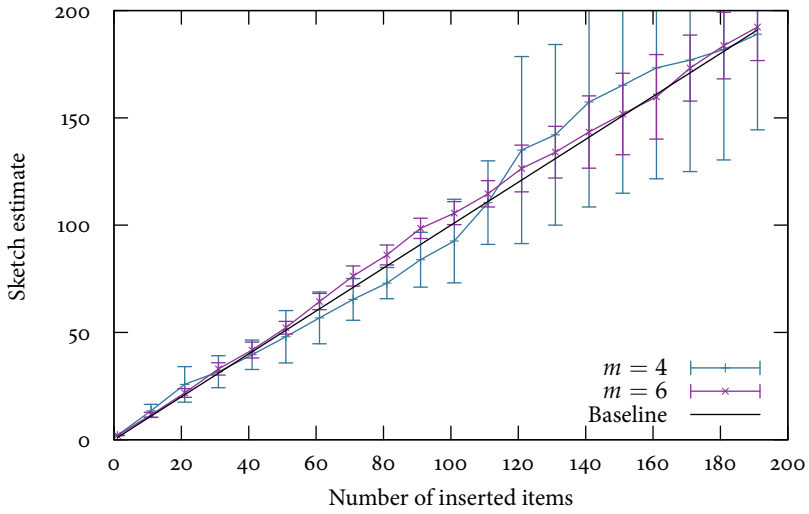


Figure 9.7: Average sketch estimate for different numbers of inserted items.

tack success only rises as the attacker controls almost all of the vehicles within a cluster.

In Figure 9.6, we show the average absolute number of sketch signatures controlled by an attacker. Here, the increase is linear. That is, if attackers control n percent of all cluster members, they control, on average, n percent of all signatures in the sketch.

9.5.3 *Hierarchical aggregation accuracy trade-offs*

The HyperLogLog data structure we use as proof of cluster size is a probabilistic mechanism. Therefore, certain error margins in the estimated cluster size are to be expected. For inter-cluster dissemination and hierarchical aggregation decisions, we use the intersection formulae shown in Equation (9.14), which combines several sketch estimates and may introduce an even larger error. Existing work by Heule et al. [81] analyzes the behavior of HyperLogLog sketches and concludes that they offer sufficient accuracy for large counts and large values for m. Because we expect smaller counts of vehicles and small values of m, we conduct a separate study to analyze the probabilistic nature of HyperLogLog sketches in our setting.

Sketch accuracy.

Figure 9.7 shows the cluster size as estimated by the sketch (y-axis) for cluster sizes between 0 and 200 vehicles, as shown on the x-axis. The standard deviation is shown for 10 different random clusters per cluster size. For $m = 4$, corresponding to 16 attached signatures, the standard deviation is high, especially for larger clusters. Setting $m = 6$ decreases the standard deviation significantly but results in 64 attached signatures. Figure 9.8 shows the estimates for the clus-

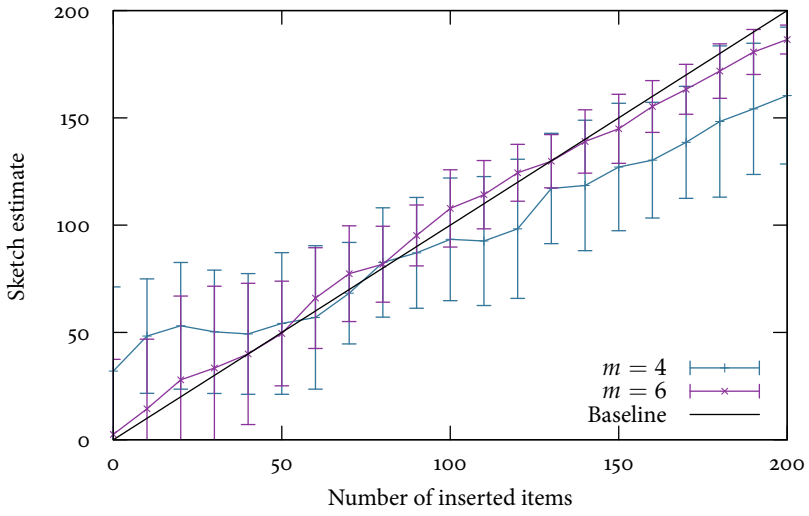


Figure 9.8: Average sketch estimate for different numbers of intersecting items.

ter intersection size for $m = 4$ and $m = 6$. Here, the values on the x-axis represent the size of the intersection between two clusters. To generate the graphs, random clusters are generated that intersect in x elements. The standard deviations are comparable to the insertion comparison. However, the line for $m = 4$ clearly shows a bias that is introduced, because the intersection formulae use three estimates and their errors accumulate. For intersection sizes between 0 and 50 vehicles the sketch with size $m = 4$ consistently over-estimates the intersection size. For larger intersection sizes, the sketch under-estimates the intersection size consistently.

Both the insertion estimates and the intersection estimates show a clear trade-off. For the smallest value $m = 4$, the error introduced by the sketch is considerable. Yet, the resulting estimate is still sufficient for integrity protection. The goal of the cluster size proof is to make it unlikely that an attacker can successfully control all attached signatures. As shown in Figure 9.5, this goal is achieved. The error margin can be compensated by increasing the cluster size threshold τ accordingly. For instance, if $\tau = 10$ is the desired value and the sketch error is 5, setting $\tau = 15$ will compensate the error.

For the intersection accuracy, the situation is different. With $m = 4$, the intersection criterion only offers a rough indication for cluster member intersection. Setting $m = 6$ improves the estimate considerably, but using 64 signatures for each cluster size proof is infeasible due to bandwidth constraints.

To achieve a better trade-off between integrity protection, bandwidth consumption, and intersection accuracy, we can change the signature selection scheme for the cluster size proof. Namely, we can only include a randomly selected subset of all signatures when creating the cluster size proof. For instance,

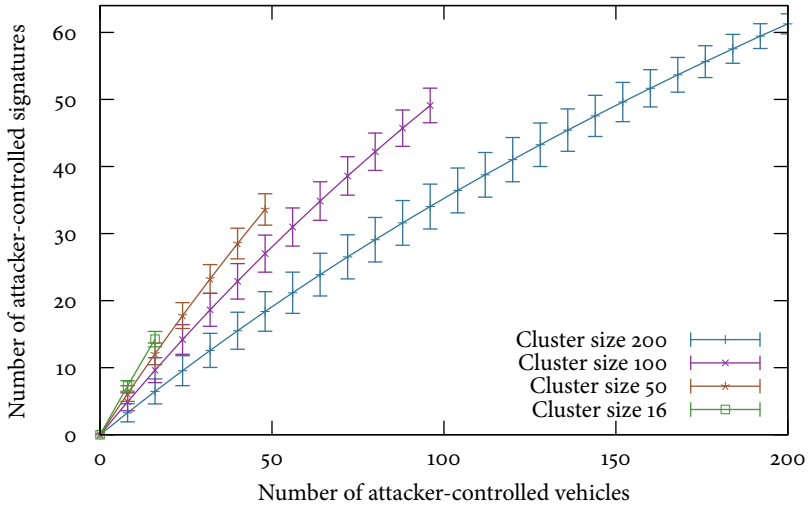


Figure 9.9: Number of attacker-controlled signatures in HyperLogLog sketch for different cluster sizes and numbers of attackers using $m = 6$.

we set $m = 6$ but only include 16 randomly selected signatures instead of 64. Since the dependency between attacker-controlled vehicles and number of controlled signatures in the sketch is linear the assumption is that the attacker success will still be negligible for larger cluster sizes. Figure 9.9 confirms this assumption. For a cluster size of 50 vehicles, the attacker needs to control, on average, 20 vehicles to be able to present 16 out of 64 signatures. For a 100 vehicle cluster, 28 vehicles need to be attacker-controlled. Thereby, we can benefit from the higher accuracy of larger sketches while maintaining the lower security overhead.

9.6 BANDWIDTH OVERHEAD ANALYSIS

For each cluster with at least τ members, the constant overhead size only depends on the number of signatures used in the cluster size proof. Since only one cluster size proof is kept even for hierarchically aggregated values, the bandwidth overhead relative to the covered stretch of road is lower if the overall traffic situation is more homogeneous. In the following, we assume that all clusters are larger than the threshold τ. Let l be the average size of a stretch of homogeneous traffic and s be the number of signatures attached to an aggregate. Further, assume that aggregate signatures are used for signing the sketch

values, as well as signing the public keys, similar to Sections 7.6 and 8.6. The overhead is then

$$O = \frac{1}{l}(s(\underbrace{32.375}_{(1)} + \underbrace{12}_{(2)} + \underbrace{1}_{(3)}) + 2 \cdot + \underbrace{32.375\,\text{bytes}}_{(4)}). \tag{9.17}$$

(1) public key
(2) key validity, id
(3) sketch value
(4) aggregate signature

Since $s \geq 16$ due to the HyperLogLog data structure, the overhead is considerable for smaller clusters. However, if traffic is homogeneous and clusters are large, the bandwidth overhead will remain constant in contrast to the mechanisms presented in Chapters 7 and 8.

9.7 ADAPTIVITY

The clustering-based security mechanism requires an average cluster size of at least τ vehicles to work optimally in terms of bandwidth consumption. Also, the cluster size proofs benefit from larger cluster sizes, because it is less likely for an attacker to control the majority of vehicles in large clusters. Therefore, it is beneficial to use the clustering mechanism in situations where the traffic along a stretch of road is homogeneous, such as traffic jams. In heterogeneous traffic situations, the mechanisms presented in Chapters 7 and 8 will provide a better trade-off between integrity protection and bandwidth consumption.

In addition, the clustering mechanism can be adapted by selecting different sketch sizes which result in different numbers of attached signatures. When an attack is more likely, more signatures can be attached to aggregates by selecting larger values for m. However, increasing m by 1 doubles the number of attached signatures due to the construction of HyperLogLog sketches. Therefore, only a subset of all sketch values can be signed, as discussed in Section 9.5.3. Selecting only a subset of signatures allows a fine-grained adaptation of the consumed bandwidth depending on current attack likelihood.

9.8 SUMMARY

Clustering is not widely used in the VANET research community due to the ephemeral node contact and due to high vehicle velocity. We designed a clustering mechanism that addresses these issues specifically and is suited for use in VANETs, because we integrate clustering decisions and aggregation decisions. Using clustering as a base technology, we achieve larger atomic units of vehicles that remain in mutual communication range for some time. Because an attacker can only compromise a limited number of vehicles, we can assume a cluster with a certain number of members as trustworthy.

To transfer this trust, which is built by mutually overhearing communication within the cluster, to neighboring clusters, we designed a bandwidth-efficient proof of cluster size. Based on the HyperLogLog data structure, our cluster size proof is especially beneficial for clusters with a large number of members. As an additional benefit, the HyperLogLog data structure allows to detect duplicates

during hierarchical aggregation. Proofs of cluster size are only performed by the last cluster in a chain of clusters during hierarchical aggregation. The presented clustering mechanism is, therefore, especially suitable for large stretches of homogeneous traffic where it outperforms the security mechanisms presented in Chapters 7 and 8.

10

This chapter is a revised an extended version of our publication [5]. Preliminary investigations for this chapter were conducted as part of a supervised master thesis [220]. The redundancy analysis (Section 10.5) is based on our publication [10].

10.1 OVERVIEW

A fundamental problem of using cryptographic security mechanisms for aggregation is that aggregation functions break cryptographic signatures. Moreover, we have to assume insider attackers can alter their own messages' content at will. Therefore we complement the schemes discussed in Chapters 7 to 9 with a scheme that is based on statistical detection heuristics. We assume that reports about a large event are observed by multiple vehicles, as discussed in Section 3.6. Other vehicles can analyze these reports and assume that information received from the majority of vehicles is correct. This statistical detection approach is implemented by clustering reported information about the same area according to the reported situation and assuming the largest cluster represents the correct information. When used alone, however, this approach is hindered by aggregation, as shown in Figure 10.1. Here, a single attacker controls the majority of incoming paths to a receiving vehicle. Therefore, we propose a combination of cryptographic signatures and statistical methods that eliminates the drawbacks of both approaches. The security mechanism is based on a flexible aggregation mechanism similar to DYN, and we use a traffic information system to concretize the mechanism's explanation.

We give an overview of the mechanism concept in Section 10.2 and discuss the main components in Sections 10.3 and 10.4. To validate the mechanism, we first introduce metrics for dissemination redundancy and demonstrate that sufficient redundancy likely exists in aggregation protocols in Section 10.5. Section 10.6 analyzes the mechanism's resilience against our attacker model. We analyze the expected overhead in Section 10.7 and discuss adaptivity parameters in Section 10.8.

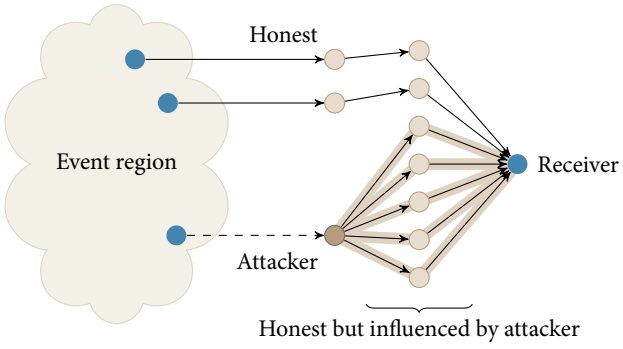


Figure 10.1: A single attacker influencing the majority of forwarding paths.

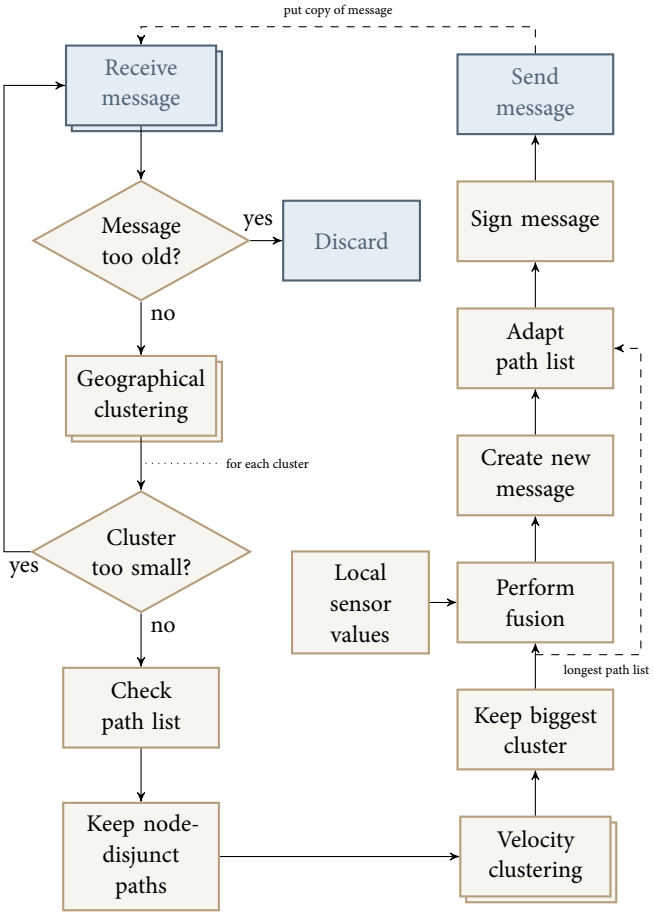


Figure 10.2: Information flow of our approach.

10.2 INFORMATION FLOW SUMMARY

We combine two main components in our mechanism. First, dissemination paths are traced and made explicit using cryptographic signatures that each forwarding or aggregating vehicle adds. These so-called forwarding path lists are then used as filter for situations where few vehicles control many forwarding paths. The remaining, filtered reports are then used as input to a data mining algorithm that groups reports by the contained information and retains the largest group as the likely correct information about an event.

Figure 10.2 describes the information flow within one vehicle from message reception to message dissemination. Each vehicle operates according to the same scheme, and the algorithm is run periodically whenever new information is received. All received messages are first added to a buffer. Messages can either contain atomic observations or aggregates. Messages pertaining to old events are discarded immediately to avoid processing outdated information.

Next, the information in the reception buffer is clustered according to the information's geographical location. The goal is to detect groups of messages that contain information about a common "event" in a confined geographical region. At this stage, the clustering is only based on locations. Hence, detected events group messages with potentially conflicting information. After event detection, the size of each cluster of messages is assessed. Only clusters with more messages than a pre-defined threshold are further processed. The threshold is necessary, because our scheme later uses a majority decision to filter conflicting information. If too few messages are available for an event, a majority decision would not produce meaningful results. Since the goal of in-network aggregation is to detect large scale events, all reports about true events are likely to pass the minimum threshold. If an event is reported by fewer vehicles than defined by the threshold, the corresponding cluster of messages is placed back into the reception buffer. Likely, more messages about the same event will be received in the future and the event will pass the threshold in the next iteration. If no more messages are received, the event will eventually be discarded as too old. As aggregation is often used to manage a large number of reports about the same event, however, we argue that it is unlikely that messages are discarded in this way.

The following steps are executed separately for each detected cluster of messages. Each message within a cluster can be regarded as a redundant source of information about the same event. To reduce possible influence of single attackers, we remove all messages from the cluster that were not received via node-disjoint paths. As a result, each insider attacker can influence at most one forwarding path from the event towards the receiving vehicle. To detect node-disjoint paths, we attach a list of the last $n_{\max}$ forwarders to each message, and only keep messages with mutually disjoint forwarder lists. We describe the details of the forwarder list, its integrity protection, and the filtering mechanism in Section 10.3.

After the "check path list" step, received messages have been clustered according to their geographical location and redundant messages received via partly overlapping forwarding paths have been filtered. However, each remaining group of messages still contains possibly conflicting information, because no clustering or filtering based on contained information has been applied yet. In the next step, we use a second clustering process to detect and filter such conflicts. The second clustering is applied separately for each previously detected event. The underlying assumption is that the traffic situation within an event area should be homogeneous. For instance, in a traffic jam each vehicle should move at most 5 km/h. Reports about much higher speeds in the same geographic area are indicators for an attack. Likewise, reports about the number of free parking spots in the same geographic region should contain similar numbers. Therefore, the second clustering mechanism groups messages about the same event by their velocity values. Following the honest majority assumption, only the biggest resulting cluster is kept and accepted as the most likely

Note that the homogeneity assumption may not hold for highways with many lanes per direction; in these cases, our mechanism could be adapted to use lane-exact location information and detect conflicts separately per lane.

traffic situation. All smaller clusters are discarded as likely attacks. The details of our clustering approach will be discussed in Section 10.4.

Once outliers have been filtered, all remaining messages about the same event will be combined to an aggregated message. This is the aggregation mechanism's fusion component. The aggregated message area can be derived from the area that was detected by the geographical clustering. Likewise, the contained velocity is the average velocity of all remaining messages. Moreover, the longest forwarder path list will be attached to the aggregated message. The aggregating vehicle adds a new entry to the path list that represents its own forwarder role. We chose the longest path list, because it offers the highest probability to detect non-disjunct nodes in the forwarding path for receiving vehicles. Finally, the aggregated message is signed by the aggregating vehicle and disseminated to neighboring vehicles using single-hop broadcast. To save bandwidth, only the aggregated messages for each event are disseminated further, and the separate redundant messages are discarded. Besides dissemination to other vehicles, the aggregated message is added back to the own reception buffer to facilitate hierarchical aggregation. In the future, other messages about different geographical areas within the same event might be received, which can be aggregated further.

10.3 EXPLOITING MULTI-PATH PROPAGATION

We already discussed the merits of multi-path propagation in Section 4.6.3. Basically, it is significantly harder for an attacker to influence multiple or all message paths. If a single message is transferred via multiple paths, the receiver can check for conflicting information received about the same event via multiple paths. But using such statistical methods alone in aggregation protocols has drawbacks. If an attacker influences few paths, the manipulations can be identified and eliminated. If an attacker is able to influence the majority of paths towards a receiving vehicle, statistical methods tend to "trust" the attacker and to blame the genuine messages to be forged, because it is more likely that there are more trustful network nodes than attackers.

Verifiable forwarding history.

To exploit node-disjoint paths, we need to make each message's forwarding history verifiable. Therefore, we introduce an integrity-protected path list, which we add to every message. Intuitively, each vehicle that forwards messages adds an identifier to a path list that is kept together with the message. The path list mechanism is comprised of two parts: an integrity protected path list creation and a mechanism to filter node-disjoint paths based on the lists. To prevent attackers from altering path lists, we use an adapted version of onion signatures, which were originally introduced by (Raya2006-efficient-secure-aggregation). The first entry e_0 in the path list, generated by vehicle v_0, is a signature on a randomly selected value $r \in \mathbb{N}$:

$$e_0 = (r, \sigma_{v_0}(r), pk_{v_0} cert_{v_0}) \tag{10.1}$$

Algorithm 10: Filtering node-disjoint paths.

```
DATA: A list of forwarder lists F = {F_1, ..., F_m}
RESULT: A subset of F containing node-disjoint paths.
K ← ∅
R ← ∅
FOREACH F_i ∈ F DO
    IF K ∩ F_i = ∅ THEN
        K ← K ∪ F_i
        R ← R ∪ {F_i}
    END
END
RETURN R
```

where $\sigma_{v_0}(r)$ is a cryptographic signature on r generated using v_0's secret key. Subsequent vehicles create their signature on the previous entry's signature rather than the random value:

$$\forall i \geq 0 : e_{i+1} = (\sigma_{v_{i+1}}(\sigma_{v_i}(\cdot)), pk_{v_{i+1}} cert_{v_{i+1}}). \tag{10.2}$$

Because each vehicle *over-signs* the previous vehicle's signature, an attacker cannot inject or remove entries in the path list. Contrary to Raya et al. [145], we keep all onion signatures rather than only the last 2 in order to facilitate path list verification. To reduce communication overhead, we define a maximum path list length $n_{\max}$. Once the entry $e_{n_{\max}}$ is added to the list, e_0 is discarded, and so forth. Note that discarding e_0 makes verifying e_1 impossible, but all other e_i where $i > 1$ can still be verified. In addition, we define a shorter list entry format that only contains a short hash of the vehicle's certificate $e'_j = h(cert_{v_j})$ instead of its signature, key and certificate; and we define a second threshold $n_{\text{sig}} < n_{\max}$. For each list entry older than n_{sig}, the short entry format is used. Hence, the complete path list produced by vehicle v_i has the following form:

$$(\underbrace{e'_{i-n_{\max}}, \ldots, e'_{i-n_{\text{sig}}-1}}_{\text{short certificate hashes}}, \underbrace{e_{i-n_{\text{sig}}}, \ldots, e_i}_{\text{full signatures}}). \tag{10.3}$$

All short entries e'_j cannot be used to detect manipulations, but they *can* be used to detect non-disjunct forwarder lists, because they are based on the vehicles' certificates.

Path list verification.

A receiver can use a list of messages pertaining to the same event together with their respective path lists to filter node-disjoint paths. First, each receiver checks whether the onion signature chains are correct and discards the message otherwise. The path list entries can then be used to derive a list of previous forwarders for each message. The short entry format e' already identifies forwarders; the long format e contains the vehicle's certificate, which can be hashed to acquire the same format. Let

$$F_i = \{f_{i,1}, \ldots, f_{i,n_{\max}}\} \tag{10.4}$$

be the list of forwarders for message i. Then Algorithm 10 takes a list of forwarder lists and returns a subset thereof that only contains node-disjoint paths. Intuitively, each forwarder list is added to a list of known nodes K and added to the result R until duplicates are found, i.e., the intersection of K and F_i is nonempty. The algorithm can be implemented with little computational overhead. If K is implemented as a hash set, the intersection check for each path list F_i takes $O(|F_i|)$. Since $|F_i| \leq n_{\max}$ has a constant upper bound in our scheme, the total runtime is $O(|\mathcal{F}|)$, that is, it is linear in the number of forwarding paths per event.

10.4 EVENT AND OUTLIER DETECTION

Our algorithm uses clustering methods to detect events and outliers. We use a centroid clustering mechanism [224], which is a hierarchical clustering mechanism. The advantage over partitioning clustering, such as the k-means algorithm, is that we do not need to predetermine the number of clusters. Compared to other hierarchical clustering methods, centroid clustering is beneficial, because it does not suffer from chaining effects.

The chaining effect describes gradual growth of clusters due to incremental adding of single elements, which leads to biased results.

We first give an abstract overview of centroid clustering, before we explain its application to event detection and outlier detection. First, the elements to be clustered are represented as *feature vectors* $\{\vec{v}_1, \ldots, \vec{v}_n\}$. We define a distance metric between two vectors:

$$d : \vec{v}_i, \vec{v}_j \mapsto x \in \mathbb{R}^+. \tag{10.5}$$

Given the list of vectors to be clustered, centroid clustering evaluates the distance metric for each pair of vectors and clusters those two with the smallest distance:

$$\min_{i,j \in \{1,\ldots,n\}, i \neq j} d(\vec{v}_i, \vec{v}_j). \tag{10.6}$$

Those two vectors are then merged into a cluster and are now represented by their centroid vector

$$\vec{c} = \frac{1}{2} \cdot (\vec{v}_i + \vec{v}_j). \tag{10.7}$$

The clustering process continues hierarchically, in each step merging the two (centroid) vectors with the minimum distance. If two clusters are merged, the new centroid vector is calculated using a weighted average, taking into account the cluster sizes as weights. The algorithm terminates when the minimum distance is larger than a defined threshold τ. The algorithm's initial step takes $O(n^2)$ time, because the distances between all elements need to be calculated. Afterwards, each step takes $O(n)$ time for finding the minimum distance and calculating the distances for the newly created cluster. The total runtime is $O(n^2)$. In our case, n is determined by the total number of messages received by a vehicle during a fixed time period. We argue that the quadratic complexity is tolerable, because information is aggregated during dissemination, keeping the number of received messages low.

Event detection

To detect events, in the first clustering process in our protocol, we use the geographical location of information as clustering criteria. Atomic information is represented as feature vectors according to its geo-coordinate. Without changing the concept, the feature vector could be extended to include driving direction in order to distinguish between different roads.

$$\vec{v}_o := (x_i, y_i). \tag{10.8}$$

For aggregates, which contain information about a geographical area, we use their center coordinate as feature vectors:

$$\vec{v}_a := (\mu(x_{\min}, x_{\max}), \mu(y_{\min}, y_{\max})) \tag{10.9}$$

where μ is the arithmetic average and $\{x, y\}_{\min,\max}$ are the area's minima and maxima in x and y direction, respectively. As distance function, we use the Euclidean distance between two items:

$$d_{\text{event}} := ||\vec{v}_i - \vec{v}_j||. \tag{10.10}$$

The threshold τ_{event} is set such that different events, i.e., different traffic situations are not merged. For instance, two traffic jams with a stretch of free-flowing traffic in between them should not be clustered.

Outlier detection

In our protocol's second clustering step, outlier detection, the goal is to filter spurious information within a set of information that has already been clustered by events. Here, spurious means that contained velocity values of some information deviate from the majority of values, as discussed in the attacker model (see Section 4.4). Hence, we use the average velocity as a feature vector for this clustering step. Since velocity is one-dimensional, the distance metric is simply the difference between two values:

$$d_{\text{outlier}} := |v_i - v_j|. \tag{10.11}$$

The threshold τ_{outlier} is set such that different traffic situations, e.g., traffic jam and free-flowing traffic, are not merged into combined clusters. Assuming an honest majority of vehicles, the biggest cluster within an event reports the correct velocity, and all smaller clusters are discarded.

Example clustering process.

Figure 10.3 shows an example clustering process represented as a dendrogram. Velocity values $v_1, \ldots, v_6$ all report different velocities about the same event. Reports $v_1, \ldots, v_4$ are from honest vehicles, and v_5, v_6 in this example are controlled by the attacker. During each clustering step, the two reports with the minimum distance are clustered. Clustering is represented by two merged lines, and the level indicates the step at which values were clustered. In the first step, v_1 and v_2 are merged. The resulting cluster is represented by the centroid of both velocities, i.e., their average (38 km/h). In the second step, v_3 and v_4 are

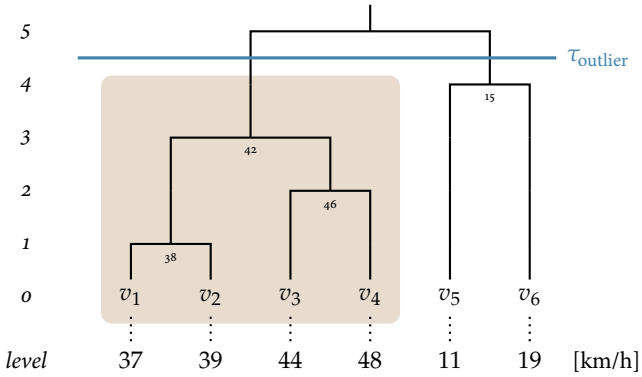


Figure 10.3: Example dendrogram for six velocity reports about the same event, of which two are outliers (v_5, v_6).

merged. In the third step, the already clustered values (v_1, v_2) and (v_3, v_4) are clustered again, forming a cluster (v_1, v_2, v_3, v_4). Normally, the process continues until only one cluster is left that contains all values. To detect outliers, we cut off the clustering process when the distance of two items is larger than τ_{outlier}. In the example, the process stops after level 4 and does not merge the attacker values v_5 and v_6 with the other values. The output of the process are two clusters (v_1, v_2, v_3, v_4) and (v_5, v_6). The biggest cluster, (v_1, v_2, v_3, v_4) is selected, and the other clusters are discarded.

10.5 REDUNDANCY ANALYSIS

To examine whether redundancy-based protection is feasible, it is a prerequisite to analyze the expected redundancy in different communication situations. Therefore, we derive two metrics that describe redundancy in multi-hop communication. And we apply those metrics to measure redundancy of dissemination in aggregation protocols in different traffic contexts. To understand the relative redundancy of aggregation, we further compare simulation results with an efficient geocast protocol and controlled flooding as a baseline.

Redundancy metrics.

We derive both metrics by analyzing the information flow graph we introduced in Section 3.6.4. The metrics we employ are

NODE-DISJOINT PATHS ($\mathcal{P}$) – the number of completely independent communication paths, which is a metric to judge the resilience against insider attackers; and

CRITICAL NODES ($\mathcal{C}$) – the number of nodes that are part of *all* paths between the message source and destination, which is a metric to judge how likely a randomly positioned attacker can successfully alter all message copies.

Figure 10.4 shows an example information flow for an aggregate A; the node-disjoint paths are highlighted. Four sources contribute the original observa-

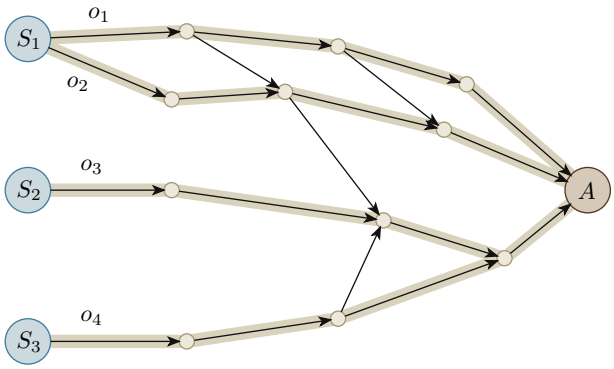


Figure 10.4: Example information flow graph with four node-disjoint paths ($\mathcal{P}$).

tions for the aggregate. Information is forwarded along a number of paths, among which 4 are node-disjoint. Calculating the number of node-disjoint paths can be done by employing maximum flow algorithms, such as Edmonds and Karp [59].

If a protocol is resilient against insider attacks, there are at least two paths that have no common nodes apart from s and d. In general, the number of node-disjoint paths ($\mathcal{P}$) characterizes how resilient a message transfer is. Given at least two node-disjoint paths, an attack by a single attacker can be detected, even if, for $\mathcal{P} = 2$, it is still undecidable which node is the attacker. For $\mathcal{P} \geq 3$, an attacker can be detected given an honest majority and additional information about the forwarding topology.

In case we have $\mathcal{P} = 1$, there is at least one node in the network that can successfully alter all messages that nodes farther away from an event receive. However, not all nodes on the node-disjoint path between s and d can attack successfully. Therefore, we calculate the number of critical nodes ($\mathcal{C}$) on the path, as well. A node is critical if its removal would disconnect the graph. In case $\mathcal{P} \geq 2$, the number of critical nodes is automatically 0. The number of critical nodes $\mathcal{C}$ indicates how likely an attacker is successful. The more nodes are in the set of critical nodes, the more likely it is that an attacker that is randomly positioned in the network is successful.

Evaluated protocols.

We apply our metrics to three different protocols. The goal is to analyze to what extent especially aggregation protocols provide enough redundancy to detect attackers in different scenarios. Also, we want to gain an understanding of the expected redundancy relative to other common dissemination protocols. We implemented representatives of the following protocol families.

BASELINE. To have a baseline, we create a graph that represents the node connectivity based on the chosen simulation parameters. This graph resembles the result of a naïve flooding with perfect packet delivery even over multiple hops. The baseline gives an estimate of the maximum achievable redundancy in a network.

GEOCAST. We use an adaptive, probabilistic gossiping protocol, namely advanced adaptive geocast (AAG) [24], as representative for the geocast protocol family. In AAG, each node determines the message forwarding probability based on the current perceived node density according to 2-hop neighborhood information. The protocol performance can be adjusted by configuring an *average reception percentage*, which states the percentage of nodes that should, on average, receive a message. In high node density scenarios, AAG uses a logistic function to automatically reduce the forwarding probability further. A *target region* can be specified that determines the area, for which an observation is relevant. For our simulations, we set the target region to the whole network, because we assume a traffic information application where all vehicles are interested in the speed of the other vehicles in the network.

AGGREGATION. We use `FIX` rather than `DYN` as aggregation scheme for our comparison. Using `FIX` provides an estimate for the lower bound of expectable redundancy in aggregation mechanisms. Using `FIX` eliminates cases where `DYN` may lead to more redundancy, because many vehicles forward information gathered in large stretches of homogeneous traffic. For calculating our metrics, we assume that a message from the source reaches the destination if the destination receives an aggregate that the source message contributed to.

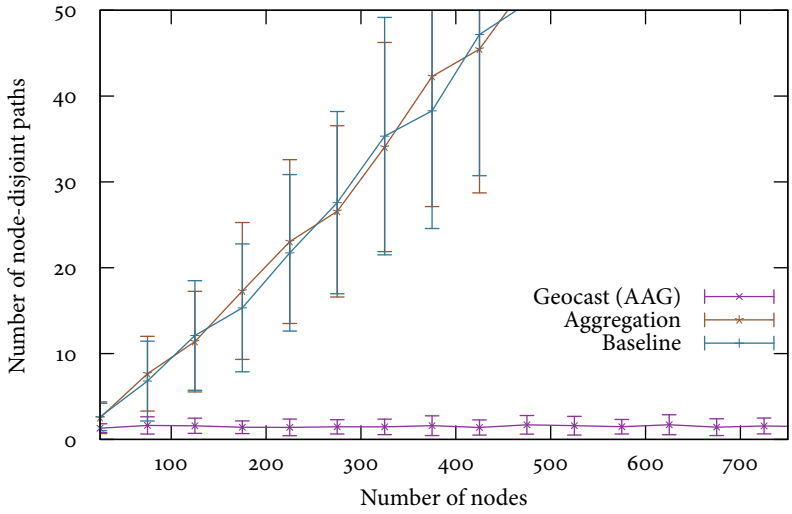
For these protocols and for each simulation setting, we calculate the redundant paths and critical nodes between 100 randomly chosen source-destination pairs in different randomly selected node placements. All other vehicles, apart from participating in the forwarding process between the source and the destination, create and disseminate messages as well. These messages are regarded as background load. They contribute to the channel load and cause possible collisions on the wireless medium, resulting in fewer paths from the source to the destination. All graphs show the average values and their standard deviations. We do not consider node mobility at this point, because we focus on single message transfers between a source and destination and can assume that the basic network characteristics, e. g., node density, remain the same during one message transfer.

We will now summarize the simulation results for common protocol parameters found in the literature. More simulation results for different parameter sets can be found in [10].

Node-disjoint paths on a highway.

Figure 10.5a shows the number of node-disjoint paths on a highway with varying node density. For the baseline and for `FIX`, $\mathcal{P}$ grows linearly in the number of nodes. For the baseline, this behavior is expected, because the graph is more connected with higher node density. `FIX` behaves similarly, because in the highway scenario there are only 10 road segments. Both protocols show a high standard deviation of $\mathcal{P}$, which is due to the varying distance of the chosen source-destination pairs. Figure 10.5b shows that the number of critical nodes $\mathcal{C} = 0$ for both the baseline and `FIX`. However, low node densities

(a) Node-disjoint paths $\mathcal{P}$



(b) Critical nodes $\mathcal{C}$

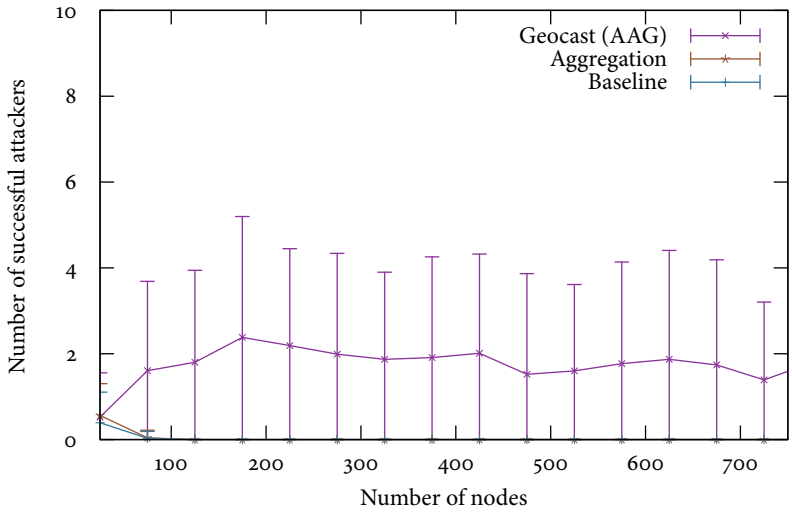


Figure 10.5: Node-disjoint paths and critical nodes for different node densities on a highway.

can be problematic. To some extent, this cannot be changed. If only few nodes are available as source of information, the redundancy is necessarily low. The results for AAG in Figure 10.5a show a much lower number of node-disjoint paths; the average over all simulation runs is 1.51, and it stays constant with growing numbers of nodes. The reason is that AAG automatically reduces redundancy by lowering the forwarding probability in high node density scenarios. Consequently, Figure 10.5b shows a higher number of critical nodes for AAG. Also, a number of critical nodes remain even in high density scenarios. While these results show that AAG reacts well to high node densities from an efficiency point of view, it is problematic from a security point of view. Even when a large number of nodes, and consequently redundant observations, is available, a cleverly positioned insider attacker will still be able to insert wrong information.

Node-disjoint paths in a city.

In the city scenario, we see similar results for the baseline and AAG protocols both in terms of node-disjoint paths and in terms of the number of critical nodes (Figures 10.6a, 10.6b). However, `FIX` performs significantly worse than in the highway case. Even for high node densities, information exchange can be attacked in some cases. The reason for this is that `FIX` needs to disseminate a much higher number of segments in the city scenario due to the larger road network. These results confirm a lesson learnt [152] from early aggregation protocols: aggregation schemes that use fixed segments do not scale with larger areas, because the number of messages that need to be disseminated still grows linearly. In contrast, schemes that adapt the aggregation areas dynamically can reduce the number of total messages, which would result in higher redundancy. Our results therefore indicate that dynamic aggregation schemes, such as `DYN`, are also favorable from a security point of view.

Drawbacks of AAG.

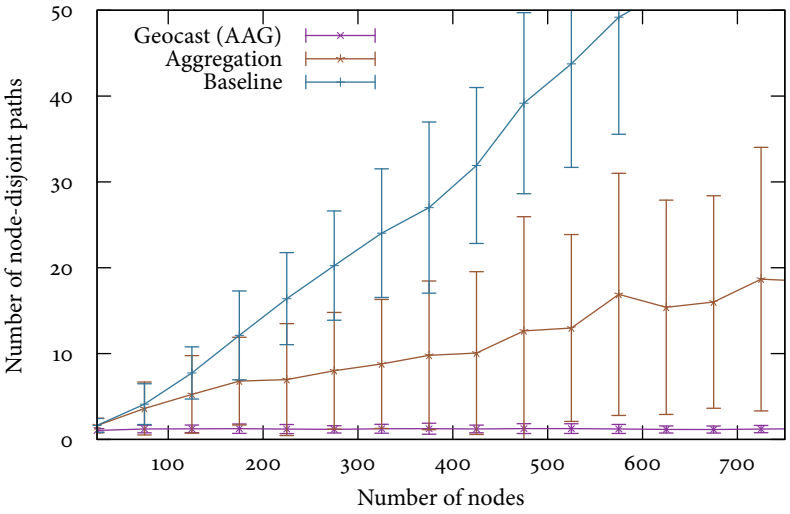
We observe that for AAG all metrics are consistent for both the city and the highway scenario, as well as for different node densities. In all settings, AAG performs almost optimal in terms of communication efficiency. However, the low redundancy due to efficient communication comes at the cost of possible attacks.

In contrast, `FIX` shows much higher values for $\mathcal{P}$. In the highway scenarios, the aggregation protocol achieves redundancy values close to the results of the baseline simulations. However, we can also see from the city scenarios that the performance of the simple aggregation protocol used is highly dependent on the total number of road segments. These results show that redundancy is a suitable basis for designing resilient aggregation mechanisms.

10.6 SECURITY EVALUATION

To assess the security aspects of our scheme, we simulate a highway scenario. Vehicles are initially distributed randomly along the highway. During the simulation, they use a car-following mobility model where each vehicle tries to reach a target speed but breaks and overtakes to avoid accidents. The simulated high-

(a) Node-disjoint paths $\mathcal{P}$



(b) Critical nodes $\mathcal{C}$

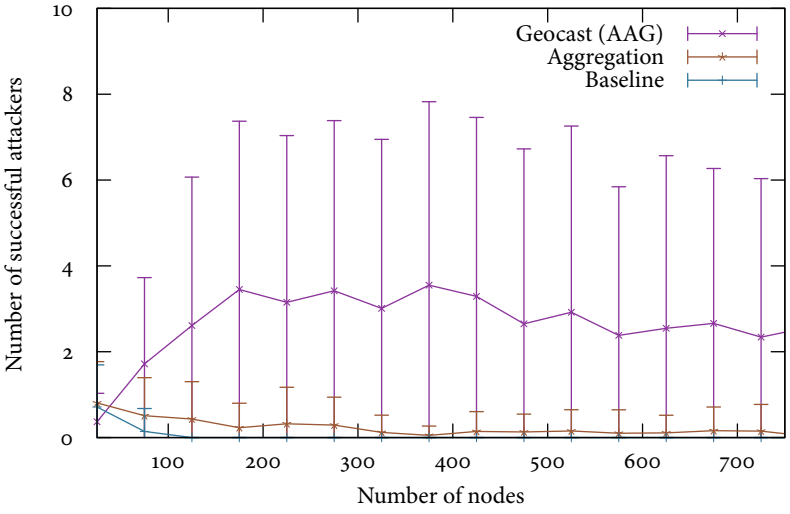


Figure 10.6: Number of node-disjoint paths and critical nodes for different node densities in a city.

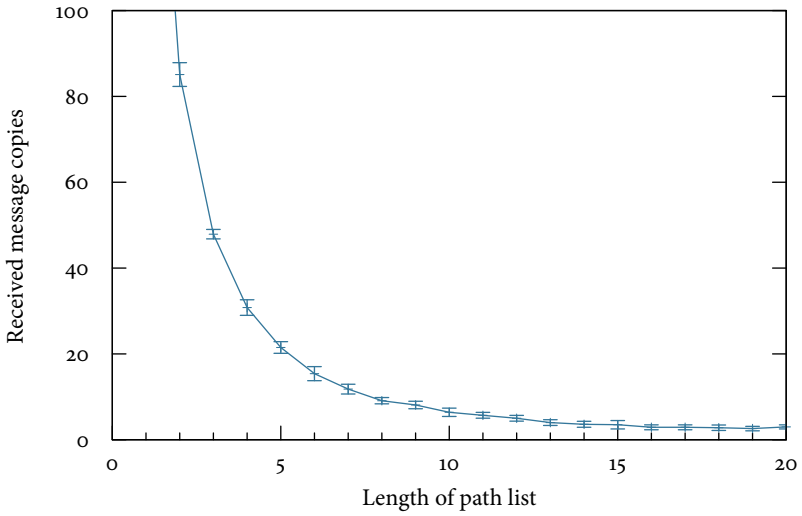


Figure 10.7: Influence of path list length.

way is 3500 meters long and has 3 lanes. The amount of vehicles varies between 100 and 250 vehicles to assess different vehicle densities.

First, we determine a suitable path list length as a trade-off between filtering message copies and overhead. Then, we analyze whether the attacker messages, after applying the path list filter and statistical outlier detection, are successfully detected and eliminated.

10.6.1 *Influence of path list length*

The path list is one of the main parts of our security mechanism. Longer path lists help in reducing redundant dissemination of messages. Moreover, long path lists make it more likely that an attacker, which may be part of many different paths, can be detected. However, the path list length also directly influences the mechanism's overhead. Each additional path list entry amounts for an extra signature and certificate, which consume considerable communication bandwidth. Therefore, the goal is to find a compromise between detecting redundant paths and bandwidth overhead.

To test the path list influence, we simulate random traffic on a highway. In an aggregation protocol, each vehicle acts as information source and sink at the same time. However, we randomly select one vehicle as *source* and one as *sink* to evaluate the path list influence between the selected pair of vehicles. The source, sink, and all other vehicles use the protocol presented in Sections 10.3 and 10.4 for message dissemination, but we disable outlier detection to isolate the path list influence. We repeat the simulation with 10 randomly chosen pairs to eliminate statistical outliers.

Figure 10.7 shows the simulation results for varying path list lengths in a network with 200 vehicles. The x-axis shows the maximum path list length $n_{\max}$ (see Section 10.3), and the y-axis shows the corresponding average number of message copies arriving at the sink, as well as the standard deviation. As expected, longer path lists reduce the total number of redundant paths. A short path list length means that vehicles may have forwarded messages but are removed from the path list shortly afterwards, which results in them forwarding the message again. For path list lengths between 10 and 20, the reduction of redundant messages is much lower. Therefore, we chose a path list length of 10 as a compromise between bandwidth usage and detecting redundancy.

Simulation results for path list lengths.

10.6.2 *Attack detection*

To assess the attack detection capabilities of our security mechanism, we simulate a traffic jam on a highway. All vehicles are standing still or driving with low velocity (≤ 5 km/h). Honest vehicles are reporting their speed accordingly and are aggregating values as described in Sections 10.3 and 10.4. The attacker behaves according to the attacker model described in Section 4.4 and implements a `CONCEAL`-type attacker. Attacker messages, therefore, contain information pertaining to free-flowing traffic. Besides manipulating own atomic observations, the attacker also creates aggregates with false speed information. To judge our protocol's attack detection, we measure the extent to which a vehicle's perceived situation on the road differs from the actual situation with a root mean square error (RMSE) metric, as discussed in Section 4.3.2.

To separate effects of our aggregation scheme and attacker influence, we simulate three different scenarios.

Different simulation scenarios.

SCENARIO 1: The *baseline scenario* represents a traffic jam with no attackers. We use it to validate our aggregation mechanism.

SCENARIO 2: The *attack scenario* shows the influence of an attacker that creates aggregates signaling free-flowing traffic. The attack is mounted on the aggregation mechanism *without* countermeasures. We use it to validate the attacker model implementation.

SCENARIO 3: The *countermeasures scenario* is the same as Scenario 2 but with countermeasures enabled. We use it to assess the performance of our security mechanisms.

To implement Scenario 2, we created a variant of our protocol that uses the same decision, fusion, and dissemination algorithms but does not check for possible attacks. That is, the path-list-based filtering (Algorithm 10) is disabled, and we set the outlier detection threshold $\tau_{\text{outlier}} = \infty$ (see Section 10.4) so that all attacker messages are merged with correct messages about the same event.

The simulation results are shown in Figure 10.8. Each scenario was simulated for 10 random vehicle distributions to eliminate statistical outliers. For each scenario, the *logarithmic* y-axis shows the achieved accuracy, which is averaged

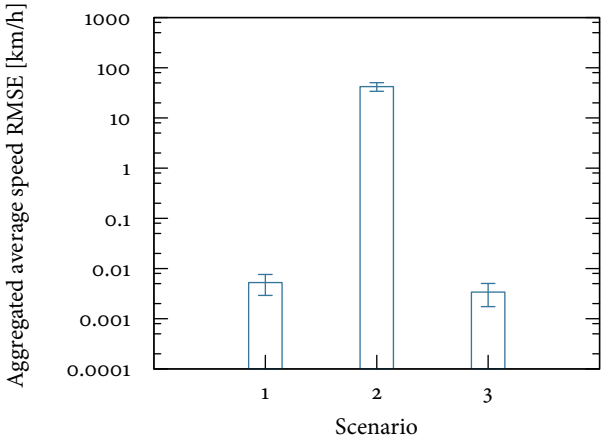


Figure 10.8: Accuracy of aggregation without attack (1), with attack (2), and with enabled countermeasures (3).

over all vehicles' world models and simulation runs. In the baseline scenario (1), the average error is 0.005 ± 0.002 km/h. When the attacker tries to disrupt the aggregation process (2), the aggregation error rises to 42.160 ± 8.216 km/h. Both the increased average error and the higher standard deviation show that the attacker is successful when no countermeasures are used. With enabled countermeasures (3), the actual situation is again correctly detected by all vehicles. The average error is 0.003 ± 0.001 km/h. Given the low standard deviation, it is likely that the attacker cannot convince any vehicles of the crafted traffic situation.

Figure 10.9 shows a more detailed version of the simulation results. Recall that the RMSE calculation is different for each vehicle, because the real vehicle speeds are compared against the aggregate speeds in the vehicle's world model. In Figure 10.9, we show the RMSE separate for each vehicle. The RMSE values are plotted on the *logarithmic* y-axis, and the x-axis represents the vehicle's positions on the road at the end of the simulation. Without an active attack (values marked as black points in the graph), the error is low for most vehicles. Few spikes above 10 km/h error are due to vehicles quickly decelerating because they approach a traffic jam, which is not represented by the aggregated values yet. The attacker successfully deceives all vehicles when no countermeasures are active, as shown by the gray lines. The spike here indicates the attacker position: the attacker amplifies the attack's effect by broadcasting a very high velocity. With enabled countermeasures (shown as red lines), RMSE values are under 1 km/h for each vehicle; the attack is not successful. Even vehicles in the attacker's direct vicinity (between the 1600 and 1700 km mark) successfully detect the attack. Moreover, the spikes that showed in the baseline scenario are filtered. This shows an additional benefit of our scheme. Besides filtering malicious attacks, the scheme filters unrealistic readings, which might indicate faulty sensors. Note that the downward trend of the red lines is due to the simu-

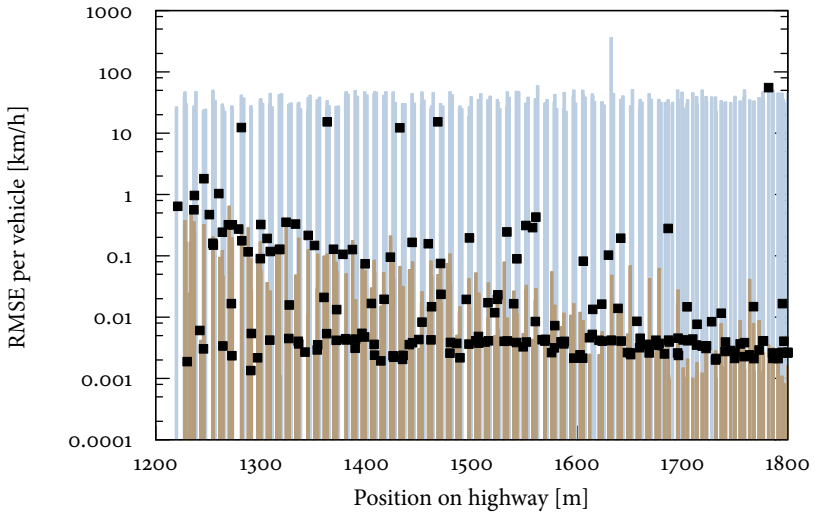


Figure 10.9: Accuracy of aggregation per vehicle without attack (■), with attack (■), and with enabled countermeasures (■).

lation scenario: to create the traffic jam, the highway is blocked at the 1800 km mark. Vehicles closer to the blocked road segment, therefore, show a more homogeneous traffic pattern throughout the whole simulation, which results in lower RMSE values.

We conclude that out security mechanism successfully detects attacks on the underlying aggregation mechanism and creates an accurate representation of the real traffic situation that fulfills the application requirements discussed in Section 4.3.2.

10.7 BANDWIDTH OVERHEAD ANALYSIS

Similar to the scheme presented in Chapter 9, the overhead of the redundancy-based security mechanism depends mostly on the traffic situation rather than on the geographic area covered by an aggregate. In addition, the configuration parameters of the scheme influence the bandwidth usage. Namely, the number of node-disjoint forwarding paths, hereafter denoted p, the number of signatures n_{sig} attached to messages, and the number of short certificates $n_{\text{max}} - n_{\text{sig}}$. Further, let l be the average size of a stretch of homogeneous

traffic. Assuming again that aggregate signatures are used, as discussed in Sections 7.6, 8.6, and 9.6, the total overhead is then

$$O = \frac{p}{l} \cdot (\underbrace{4}_{(1)} + n_{\text{sig}} \cdot (\underbrace{32.375}_{(2)} + \underbrace{12}_{(3)}) + (n_{\max} - n_{\text{sig}}) \cdot \underbrace{4}_{(4)} + 2 \cdot \underbrace{32.375\,\text{bytes}}_{(5)}). \tag{10.12}$$

(1) initial random value
(2) public key
(3) key validity, id
(4) id hash
(5) aggregate signature

The overhead is mainly influenced by the fraction p/l, which reflects the number of redundant paths p per homogeneous stretch of traffic with length l. In addition, the overhead is determined by the path list length $n_{\max}$. The overhead is, therefore, low especially for homogeneous traffic situations that result in large values for l, and and it is low for short path list lengths $n_{\max}$.

10.8 ADAPTIVITY

The redundancy-based scheme can be adapted by selecting different configurations for the path list length, as well as by altering the dissemination redundancy. If an attack is more likely, it is beneficial to keep longer path lists to make detection of attackers more likely. In addition, the number of full entries e and short entries e' can be adapted based on attack likelihood. Short entries allow to identify vehicles within a forwarding path but they do not contain signatures. Therefore, an attacker may be able to alter short path list entries to alter the information flow. Hence, more full entries should be kept in the integrity-protected path list when an attack is more likely.

In addition, the number of redundant forwarding paths can be adapted depending on attack likelihood. Most aggregation mechanisms use an opportunistic dissemination mechanism, which leads to high redundancy and large numbers of node-disjoint forwarding paths. However, redundant forwarding paths directly correlate with higher bandwidth usage. Therefore, when attack probability is low, an efficient broadcasting mechanism, such as advanced adaptive gossiping [24], can be used during the forwarding phase of aggregates.

10.9 SUMMARY

Our protection is based on the observation that the opportunistic dissemination in vehicular networks leads to information dissemination using multiple paths from a source to a destination. Likely, an attacker cannot control all of these paths, but depending on network topology, the attacker can influence the majority of incoming paths to a destination. Therefore we built a filtering mechanism that reduces information dissemination to node-disjoint paths, which reduces attacker influence. After filtering, n attackers can control at most n incoming paths to destination vehicles. In a second step, we combine our filtering mechanism with a clustering to detect conflicting information and correct it as long as the majority of incoming information is honest.

Compared to existing mechanisms, our proposal exhibits limited overhead on the wireless channel while maintaining security against common insider attacks. The only additional overhead of our mechanism is the path list, which contains a fixed number of signatures. While our mechanism requires some dissemination redundancy, we argue that such redundancy is typically already present in aggregation protocols. The reason is that the dissemination component often uses simple controlled flooding patterns rather than explicitly routing messages, as discussed in Section 3.5.2. Our proposal shows the potential of data-consistency based attack detection for in-network aggregation. Because data-consistency mechanisms require little overhead, they allow for secure aggregation protocols that still scale to large dissemination regions.

11

11.1 OVERVIEW

In Chapters 7 to 10, we presented a number of different mechanisms to make in-network aggregation resilient against attackers. The aim is to prevent attackers from altering the aggregated information arbitrarily – as defined in Section 4.3.2. At the same time, the overhead due to the additional information added by security mechanisms should be limited. In particular, the overhead should be sub-linear in the number of participants and the geographic region covered by the aggregate.

The evaluation of the proposed schemes in terms of achieved security and imposed overhead showed, however, that each single mechanism imposes certain trade-offs and fails to keep the overhead sub-linear. That means, that each mechanism would not scale to infinite numbers of aggregate participants or infinitely large geographic regions. Yet for practical applications, infinite scaling may not be necessary. As discussed in Section 7.6.2, covering a stretch of several kilometers length on a highway, or a smaller area in a city may be enough to fulfill application requirements.

Intrinsic properties of building blocks.

Also, being linear in the number of participants is intrinsic to the security building blocks that are used. In different variations, all mechanisms create proofs that a certain number of participants agree to an aggregated value. The ways in which the participants are selected differ, as do the metrics for agreement. To convey the number of agreeing participants to receiving vehicles, it is necessary that a number of mutually different identifiers is presented. The receiver can only validate the number of proof participants if it can distinguish the identifiers. This requirement is the reason that aggregate signature schemes *do* aggregate signatures but *do not* aggregate certificates or public keys, as discussed in Section 4.6.1, and it aligns with Bhaskar et al.'s results [28].

But even covering smaller areas is not feasible with a naïve security approach. Common to all mechanisms presented so far is that they lower the security overhead using a combination of two techniques:

WITNESS SELECTION All mechanisms select a certain subset of participants to contribute to the attached integrity protection. For instance, in Chapter 7, signed atomic observations that are distributed equally in the aggregate's geographic region are selected. In Chapters 8 and 9, the subset is determined by the result of a hash function, and in Chapter 10, the subset is determined by the message forwarding history.

WITNESS COMPRESSION In addition to selective inclusion, all mechanisms compress the integrity protection information as much as possible using identity-based cryptography in general, and IBAS in particular.

Table 11.1: Overview of mechanism overhead

Mechanism	Overhead
Selective attestation (Ch. 7)	Linear in geographic area.
Multi-signatures (Ch. 8)	Linear in geographic area.
Clustering (Ch. 9)	Constant when vehicle density is high, linear in geographic area otherwise.
Redundancy (Ch. 10)	Constant in path list length; linear in the number of redundant paths.

Besides the selection criteria for attestation data, the mechanisms differ in the requirements on the traffic situation for optimal operation, as shown in Table 11.1. For instance, clustering exhibits constant overhead in dense traffic scenarios, whereas selective attestation and multi-signatures may have high overhead in dense traffic situations where traffic is likely homogeneous and leads to large geographic areas covered by aggregates. Also, we already identified a number of adaptation parameters for each mechanism. Using these parameters, the mechanisms can be configured to implement different trade-offs between bandwidth usage and achieved resilience. In addition, different mechanisms are suited for different contexts and traffic situations. In this chapter, we present a framework that integrates the presented mechanisms. The framework provides

For a complete discussion of adaptivity parameters, see Sections 7.7, 8.7, 9.7, and 10.8.

- a generic representation of mechanism output using subjective logic (Section 11.2.1) and
- means to combine outputs of several mechanisms on the same aggregate using subjective logic operators (Sections 11.2.2 and 11.2.3).

Adaptation dimensions.

Based on the generic framework, we implement mechanism adaptation in two dimensions:

TRAFFIC CONTEXT ADAPTATION Some mechanisms – such as the clustering approach (Chapter 9) – work well in homogeneous traffic situations while others – for instance, the selective attestation (Chapter 7) are better applicable to heterogeneously moving traffic. In Section 11.3, we discuss the components required to implement traffic context adaptation. Namely,

- a metric for traffic homogeneity that can be derived from already aggregated information without additional communication overhead (Section 11.3.1),
- criteria for selecting mechanisms based on the traffic homogeneity metric Section 11.3.2, and
- transition strategies between different mechanisms and representations for partial proof data (Section 11.3.3).

ATTACK LIKELIHOOD ADAPTATION All mechanisms introduce trade-offs between achieved resilience and used bandwidth. In Section 11.4, we discuss how our framework can be used to lower the bandwidth overhead when attacks are unlikely while providing high resilience when attacks are likely. Attack likelihood adaptation is implemented by

- deriving an attack likelihood metric that is based on the subjective logic opinion representations of different mechanism outputs (Sections 11.4.1 and 11.4.2) and
- means to adapt mechanisms using the adaptation parameters we discussed in each mechanism chapter (Section 11.4.3).

In the following, we first discuss the abstract framework for mechanism combination in Section 11.2 before presenting specific adaptation strategies for traffic context and attack likelihood.

11.2 MECHANISM COMBINATION

Each security mechanism evaluates aggregates – alone or in combination with other aggregates – and results in a value that reflects the confidence in the aggregate's correctness. These confidence values can be used to ignore likely incorrect information or give more weight to information with a high confidence value. To combine several detection mechanisms, it is necessary to design a framework that provides a way to express this mechanism output in a generic way and to combine output from several mechanisms. In addition, confidence values may need to be merged when multiple aggregates are aggregated hierarchically.

Extending our confidence fusion framework.

In Section 7.3, we discussed a mechanism for confidence fusion that uses fuzzy reasoning to combine output of multiple data consistency checks performed by one mechanism. We will now extend this framework to allow combination of several mechanisms, as well as more flexible expression of mechanism results using subjective logic. The extended framework is based on our publication [1], which discusses the use of subjective logic for misbehavior detection in VANETs.

Figure 11.1 shows the framework architecture. All received atomic observations and aggregates are evaluated by one or more security mechanisms. Mechanisms can be active or inactive depending on current context and traffic situation. The mechanisms' results are then represented as a so-called opinion, which represents confidence in the correctness of received information. If information is evaluated by multiple mechanisms, their opinions are merged. Likewise, opinions on several items that are merged by the Fusion component are merged. During Dissemination, the security mechanisms are again used to attach the correct proof data. Other vehicles again use this proof data to evaluate the received information.

Difference between proof data and opinions.

Note the distinction between *proof data* and *opinions*. Proof data is any information that is attached to aggregates by the security mechanisms and sent

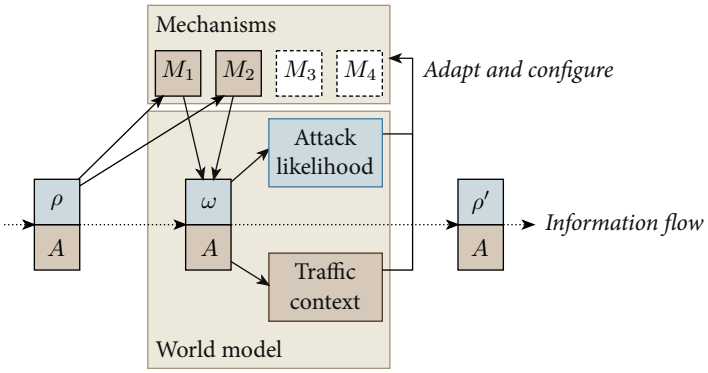


Figure 11.1: Framework architecture overview. Each enabled mechanism M_i evaluates incoming aggregates A, resulting in a merged opinion. Opinions and aggregates influence current attack likelihood and traffic context, respectively. During dissemination, aggregates are again protected with proof data.

to other vehicles. Examples are cryptographically signed atomic observations, multi-signatures on aggregates, signed forwarding paths, and signatures on FM sketches. Proof data is used by security mechanisms to evaluate information received from other vehicles. By evaluating the proof data, an opinion is created that represents the confidence in the correctness of information. Locally, information can be handled using this opinion. Whenever information are merged, the opinions are merged, too. The advantage of using opinions locally is that they abstract from mechanism specificities and enable a generic way to handle information. However, opinions are usually not disseminated to remote vehicles. Since the security mechanisms presented in Chapters 7 to 10 operate information-centric, they do not create trust in vehicles but only confidence in information. Information confidence is preferable over vehicle trust, because vehicles may not interact often enough to establish trust relationships. Hence, opinions cannot be used by remote vehicles, because using them would imply trust in the opinion holder. Rather, the correct proofs and proof data are (re-)assembled before dissemination.

11.2.1 *Proofs and opinions*

Let $A = (\mathcal{L}, \mathcal{T}, v, q_1, \ldots, q_n)$ be an aggregate consisting of a locator, primary value, and auxiliary values, as defined in Definition 5. We call all additional data that is attached to the aggregate by a security mechanism *proof data*. Aggregates may be accompanied by multiple proofs if multiple security mechanisms are used in parallel. And proofs may only attest to parts of the aggregate, which are identified by a locator that represents a specific spatio-temporal region, which

Table 11.2: Example Subjective Logic Opinions

Opinion	Interpretation
$(1, 0, 0, a)$	Binary logic TRUE.
$(0, 1, 0, a)$	Binary logic FALSE.
$(0, 0, 1, a)$	Total uncertainty.
$(0.5, 0.5, 0, a)$	Dogmatic opinion with probability $p = 0.5$.

Based on Jøsang [219, Tab. 4.1].

may be a subset of the aggregate's spatio-temporal region. Hence, the format of an aggregate and its proof data is:

$$(A, \rho_1^{(\mathcal{L}_1, \mathcal{T}_1)}, \ldots, \rho_n^{(\mathcal{L}_n, \mathcal{T}_n)}), \tag{11.1}$$

where ρ_i is the proof created by the i-th mechanism and $(\mathcal{L}_i, \mathcal{T}_i)$ is the corresponding locator. If $(\mathcal{L}, \mathcal{T})$ is the locator of the aggregate A, then $\forall i : \mathcal{L}_i \subseteq \mathcal{L}, \mathcal{T}_i \subseteq \mathcal{T}$.

When the aggregate is received, each security mechanism defines a function that maps its proof data to an opinion:

$$\begin{aligned} M_1 &: \rho_1^{(\mathcal{L}_1, \mathcal{T}_1)} \longmapsto \omega_1, \\ &\vdots \\ M_n &: \rho_n^{(\mathcal{L}_n, \mathcal{T}_n)} \longmapsto \omega_n. \end{aligned} \tag{11.2}$$

For opinion representation, we use subjective logic, introduced by Jøsang [94], which is an extension of classic Boolean and probabilistic logic that allows to express belief and disbelief under uncertainty. The format of an opinion is

$$\omega = (b, d, u, a) \tag{11.3}$$

with $b, d, u, a \in [0, 1]$ and $b + d + u = 1$. Here,

- b is the belief in the correctness of the aggregate,
- d is explicit disbelief in the correctness,
- u represents the mechanism's uncertainty, and
- a is the atomicity, also called base rate, which represents a priori probability of correctness, for instance, due to current general attack likelihood.

Table 11.2 shows exemplary subjective logic opinions and their interpretation. Note in particular, that subjective logic allows to distinguish between total uncertainty and equal belief and disbelief with no uncertainty.

Benefits of subjective logic.

We choose this subjective logic opinion representation, because it represents the outcome of our mechanisms well. Independent of the specific implementation of proof data, each mechanism uses the created proofs to conclude belief

or disbelief in the correctness of aggregates. In addition, the mechanism's results may be accompanied with uncertainty. If required, opinions can be transformed into a single confidence value between 0 and 1 by calculating their expected value:

$$E(\omega) := b + a \cdot u. \tag{11.4}$$

As shown in Equation (11.1), some mechanisms may only evaluate a proof on a smaller spatiotemporal region $(\mathcal{L}_i, \mathcal{T}_i)$ than the whole aggregate region. In those cases, the mechanism's proof evaluation function M_i should reflect the smaller region with a higher uncertainty u_i of the resulting opinion ω_i.

EXAMPLE The selective attestation mechanism discussed in Chapter 7 selects a subset of signed atomic reports as proof data. The number of attached reports, as well as their geographic distribution in the aggregate, are determined by the Security parameter S. In addition, the velocity values contained in the atomic reports can be correlated with the aggregate's claimed velocity to detect attacks. As a result, three quantitative parameters can be mapped to the opinion:

1. the number of actually attached reports compared to the expected number of reports,
2. the extent to which the attached reports correspond to the grid induced by S (denoted Δ in Figure 7.2), and
3. the difference between average velocity and atomic report velocity, as defined in Equation (7.5).

Let $A \in [0, 1]$ be the fraction of attached reports (parameter 1), $B \in [0, S]$ the grid correspondence (parameter 2), and $C \in [0, v_{\max}]$ the velocity difference (parameter 3). Here, $v_{\max}$ is the road's speed limit. The mechanism's proof data can be mapped to an opinion as

$$\omega' := \begin{cases} b' := A \\ d' := \frac{C}{v_{\max}} \\ u' := B/S \end{cases}. \tag{11.5}$$

We chose the fraction of attached atomic reports A to represent explicit belief, because attaching all underlying atomic reports would correspond to a situation without any aggregation. If all atomic reports would be available, attackers could only alter their own atomic reports rather then the aggregate itself to influence the perceived situation. As aggregation decisions should find homogeneous stretches of traffic, we treat C, the deviation of the attached atomic reports' velocity from the average, as a sign for explicit disbelief in aggregate correctness. The deviation C is normalized using the maximum allowable speed. Finally, the uncertainty is influenced by how uniform the distribution of attached atomic reports is. The more skewed the distribution, the higher the uncertainty.

The opinion ω', however, violates the summation requirement that $b + d + u = 1$. In addition, the mechanism's overall weight when being combined with other mechanisms should be expressed. To do this, we define a static parameter $\Psi \in [0,1]$ that expresses the overall weight that this mechanism should be given. For instance, $\Psi = 0.75$ means that the maximum belief expressed by the mechanism should be 0.75, and consequently, the mechanism's minimum uncertainty should be 0.25. The selective attestation is based to a large extent on the maximum number of cryptographic keys that an attacker can control. Therefore, Ψ can be high. Other mechanisms, such as the redundancy-based mechanism discussed in Chapter 10 require a lower Ψ value, because they rely on statistical information.

Taking into account the summation requirement and Ψ, the opinion mapping becomes

$$\omega := \begin{cases} b := (\Psi - u') \max(b', 1 - d') \\ d := (\Psi - u') \max(d', 1 - b') \\ u := 1 - \Psi + u' \end{cases} . \tag{11.6}$$

11.2.2 *Opinion evaluation and merging*

Once proof data is locally converted to opinion, these opinions can be used to reason about information. For instance, information that is associated with a high disbelief can be ignored by applications. Or information with high belief can be prioritized in further dissemination. In addition, subjective logic specifies a number of operators that can be used on opinions. Jøsang [219] provides an overview of subjective logic operators. We use these operators in two scenarios.

1. Multiple security mechanisms may be active at the same time and evaluate the same aggregate. As a result, multiple opinions about the same aggregate may exist that need to be merged.

2. Several aggregates may be selected to be merged by the Decision component. Then, the corresponding opinions need to be merged as well to result in an opinion about the merged aggregate.

Different reasons for opinion merging.

In the first case, all mechanisms observe the same aggregate, but they use different proof data and other properties of the aggregate to evaluate its correctness. In case some mechanisms are more confident than others, they should express this using different uncertainty ratings. Also, mechanisms can explicitly state disbelief in aggregate correctness due to the opinion format. Therefore, the combination of several mechanisms should average their output, taking into account their uncertainty as weights, as well as maintaining expressed disbeliefs. Therefore, we chose Jøsang et al.'s averaging fusion [95] for combining opinions of several mechanisms about the same aggregate.

DEFINITION 14 (Opinion averaging fusion, denoted by $\oplus$) Let

$$\omega_i = (b_i, d_i, u_i, a_i) \tag{11.7}$$

with $1 \leq i \leq n$ be opinions from n security mechanisms on the aggregate A. Then their averaging fusion $\omega_1 \oplus \omega_2$ is defined as [95]:

$$\omega : \begin{cases} b = \frac{\sum_{i=1}^{n}(b_i/u_i)}{\Delta+n}, \\ d = \frac{\sum_{i=1}^{n}(d_i/u_i)}{\Delta+n}, \\ u = \frac{n}{\Delta+n}, \\ a = \frac{\sum_{i=1}^{n} a_i}{n}, \end{cases} \tag{11.8}$$

where

$$\Delta = \sum_{i=1}^{n}(b_i/u_i) + \sum_{i=1}^{n}(d_i/u_i). \tag{11.9}$$

Intuitively, the calculation averages belief and disbelief values, using uncertainty values as weights, as shown in Equation (11.9). Once the different mechanisms' opinions are merged, the resulting opinion is attached to the aggregate and used locally to express confidence in its correctness.

Inter-relation with fusion component.

Due to hierarchical aggregation, the Decision component may, at a later point in time, decide to merge two aggregates using the Fusion component, which is expressed by the second case. In this case, we do not use averaging fusion but a different subjective logic operator. When two aggregates are merged, the confidence in their correctness may differ. If the difference is too high – e. g., $E(\omega_1) = 0.9$ and $E(\omega_2) = 0.1$, it may be used by the Decision component to circumvent merging those aggregates at all. In cases where the confidence differs by a lower amount, the combined confidence should be determined by the lower of confidence value. Otherwise, an attacker may be able to amplify the attack by having falsified aggregates merged with correct aggregates and benefiting from the higher average confidence. Therefore, we choose Jøsang and McAnally's multiplication operator [96] for merging opinions of different aggregates selected for fusion.

DEFINITION 15 (Opinion multiplication, denoted by $\circ$) Let

$$\omega_1 = (b_1, d_1, u_1, a_1) \tag{11.10}$$

and

$$\omega_2 = (b_2, d_2, u_2, a_2) \tag{11.11}$$

be opinions on aggregates A_1 and A_2. Then their multiplied opinion $\omega_1 \circ \omega_2$ on $A_1 \bowtie A_2$ is defined as [96]:

$$\omega_1 \circ \omega_2 : \begin{cases} b = b_1 b_2 + \frac{(1-a_1)a_2 b_1 u_2 + a_1(1-a_2)u_1 b_2}{1-a_1 a_2}, \\ d = d_1 + d_2 - d_1 d_2, \\ u = u_1 u_2 + \frac{(1-a_2)b_1 u_2 + (1-a_1)u_1 b_2}{1-a_1 a_2}, \\ a = a_1 a_2. \end{cases} \tag{11.12}$$

As can be seen in Equation (11.12), the belief values are multiplied, meaning that the resulting belief $b \leq \min(b_1, b_2)$. Likewise, disbelief is cumulated to ensure that the more negative opinion is favored when two aggregates are merged. Again, the merged opinion is used locally to further evaluate the combined aggregate.

11.2.3 *Proof merging*

In our framework, opinions are used only for local evaluation of aggregates. Whenever aggregates are to be disseminated to other vehicles, they need to be accompanied by corresponding proof data. How proof data is created depends on the specific security mechanisms used. If the aggregate was not modified by the vehicle, the original proof data can be re-attached and disseminated. If the aggregate was merged with other aggregates by the Fusion component, a new proof pertaining to the merged aggregate needs to be created.

Since merging of proofs is mechanism-dependent, each security mechanism defines a function M', which takes two aggregates and their corresponding proof data and creates a proof for the merged aggregate. Let A_1 and A_2 be aggregates and $\rho_j^{A_i}$ be the proof data attached by mechanism j to A_i. Then

$$\begin{aligned} M'_1 &: \rho_1^{A_1}, \rho_1^{A_2} \longmapsto \rho_1^{A_1 \bowtie A_2}, \\ &\vdots \\ M'_n &: \rho_n^{A_1}, \rho_n^{A_2} \longmapsto \rho_n^{A_1 \bowtie A_2}. \end{aligned} \tag{11.13}$$

As for merging opinions on different aggregates, the resulting proof will result in lower confidence if at least one of the original proofs resulted in lower confidence. Therefore, it will hold that

$$M_i\left(\rho_i^{A_1}\right) \circ M_i\left(\rho_i^{A_2}\right) \approx M_i\left(M'_i\left(\rho_i^{A_1}, \rho_i^{A_2}\right)\right). \tag{11.14}$$

That is, a mechanism's M and M' functions should be implemented such that the transformation of separate proofs into opinions and the transformation of the merged proof into an opinion are commutative.

11.2.4 *Summary*

The presented framework provides a mechanism to evaluate aggregates and atomic observations using multiple security mechanisms at the same time. Also, it enables merging of several opinions on aggregates when they are merged by the Fusion component. The framework foresees that not all security mechanisms are active at the same time. Also, the opinions on all known aggregates, which are stored in the world model, can be used to adapt security mechanisms to current attack likelihood. In the following, we will discuss enabling and disabling of mechanisms depending on traffic context, and we will discuss how opinions can be leveraged to adapt mechanisms.

11.3 TRAFFIC CONTEXT ADAPTATION

The presented security mechanisms are each particularly effective when used in specific traffic situations and are less effective in other situations. For instance, the clustering mechanism presented in Chapter 9 requires a certain minimum cluster size τ and operates most effectively when cluster sizes are larger than τ. On the other hand, the attestation mechanism from Chapter 7 can cope with arbitrary traffic situations, but introduces significant security overhead for aggregates about large geographic regions. Likewise, the multi-signature-based mechanism (Chapter 8) adapts to heterogeneous traffic but incurs large overhead when many vehicles contribute to an aggregate. In Chapter 10, we require high redundancy of received messages for likely detection of falsified aggregates.

Mechanism suitability.

Generalizing those specific requirements, we argue that some mechanisms are better suited for *homogeneous* traffic situations – i. e., many vehicles with similar movement patterns – whereas other mechanisms are better suited for *heterogeneous* traffic situations – i. e., possibly few vehicles with different movement patterns. In the following, we will derive a metric for traffic homogeneity that is based on properties of existing aggregation mechanisms and can be derived in a decentralized way without extra communication overhead. The traffic homogeneity metric can be used to enable or disable security mechanisms depending on context. However, switching the combination of active mechanisms can never be a clear cut in a setting as dynamic as VANETs. Therefore, we will follow up our metric and mechanism selection discussion with an approach to gradually enable and disable mechanisms, maintaining partial proof data on aggregates. This partial proof data can be used together with our framework, as discussed in Section 11.2.2, to derive merged confidence values for aggregates.

11.3.1 *Traffic homogeneity metric*

To select security mechanisms suitable for the current traffic situation, we need to define a metric that distinguishes different traffic situations. In particular, some security mechanisms (e. g., clustering, Chapter 9) depend on high traf-

fic homogeneity. Ideally, the metric should be easy to calculate and not incur extra communication overhead besides the information already exchanged for aggregation. Moreover, we want to use the metric to determine which security mechanisms should be enabled and disabled. Therefore, vehicles should be able to calculate the metric locally but arrive at the same result as their surrounding vehicles.

Deriving homogeneity form existing information.

Given available sensor readings, traffic homogeneity can be determined using the current velocity and geographic location of surrounding vehicles. The more similar velocities of large groups of vehicles are, the higher the traffic homogeneity. Note that this definition correlates to the Decision component algorithms of flexible aggregation mechanisms. In Section 3.4.3, we argued that only similar information should be aggregated, which is implemented by the Decision component as discussed in Section 3.6.3. Besides merging velocity and location of atomic observations, aggregation mechanisms also count the number of observations that contributed to an aggregate. If many atomic observations were selected for aggregation, and the resulting aggregate, therefore, has many contributors, the underlying traffic situation is likely homogeneous. Therefore, we use the average current number of contributors to aggregates about the surrounding area as a metric for traffic homogeneity. An additional benefit is that this metric can be derived from existing aggregation information without additional communication overhead. Also, deriving the homogeneity metric using information from our individual resilient aggregation mechanisms means that it is difficult for attacker to influence the metric. To do so, attackers would need to successfully attack at least one of the individual security mechanisms.

Let $\mathcal{W} = \{A_1, \ldots, A_n\}$ be the vehicle's world model containing a number of aggregates. Without loss of generality, let the data structure of each aggregate be

$$A_i = (\mathcal{L}_i, \mathcal{T}_i, v_i, \ldots, c_i), \tag{11.15}$$

where c_i is the number of participants. Further let

$$d : \mathcal{L}_1, \mathcal{L}_2 \longmapsto \delta \in \mathbb{R}^+ \tag{11.16}$$

be a distance metric that measures the distance of two aggregate locators $\mathcal{L}_1$ and $\mathcal{L}_2$ and let L be the current position of the vehicle. Then we define the current traffic homogeneity as

$$\mathcal{H} := \frac{1}{|S|} \sum_{A_i \in S} c_i, \tag{11.17}$$

$$S := \{A_i \in \mathcal{W} : d(A_i, L) \leq D\}, \tag{11.18}$$

where $D \in \mathbb{R}^+$ is the maximum distance in meters to the own vehicle at which an aggregate should still influence the average traffic homogeneity. Counting all surrounding aggregates with the same weight ensures that geographically

close vehicles will independently calculate similar traffic homogeneity values and derive the same security mechanism combination.

In addition to aggregate size, we use the number of vehicles within 1-hop communication range to determine traffic homogeneity. This additional metric reflects that the clustering mechanism discussed in Chapter 9 only clusters vehicles within 1-hop communication range. The neighborhood size can be determined without additional communication overhead, because all aggregation mechanisms are built on periodic broadcast beacons between vehicles. These beacons can be used to continuously update a neighborhood table. We define the combined traffic homogeneity as the maximum of the vehicle neighborhood size and the aggregate-based homogeneity $\mathcal{H}$.

11.3.2 *Mechanism selection*

Three of the security mechanisms presented in Chapters 7 to 10 depend on traffic homogeneity. Namely, the selective attestation mechanism (Chapter 7) can be used interchangeably with the multi-signature-based mechanism (Chapter 8) whenever the traffic situation is heterogeneous and traffic density is sparse. Due to the higher number of signatures used in the multi-signature-based mechanism, it lends itself to situations with higher chance for an attack, which we will discuss in Section 11.4. The clustering mechanism (Chapter 9) is well suited for situations with homogeneous traffic. In the following, we will describe how the traffic homogeneity metric $\mathcal{H}$ can be used to select between the selective attestation and clustering mechanism based on their bandwidth overhead analyses in Sections 7.5 and 9.6. We describe the selection mechanism exemplarily for these two mechanisms, but the same considerations apply to selecting between the multi-signature-based mechanism and the clustering mechanism, as well.

In Section 7.5, we argued that the bandwidth consumed by the selective attestation mechanism depends on the security parameter S and the geographical area covered by an aggregate. The bandwidth overhead of the clustering mechanism, however, depends on the constant choice of the number of sketches m and the percentage of clusters that have at least τ members. Both the clustering mechanism, as well as the Decision component of `DYN`, which is used by the selective attestation mechanism rely on velocity difference as the main factor for deciding whether to join a cluster and whether to merge aggregates, respectively. Therefore, the traffic homogeneity metric directly indicates whether to use clustering or selective attestation depending on the current traffic situation.

Namely, whenever $\mathcal{H} \geq \tau$, clustering will be able work work most efficiently, because all clusters are large enough to convince other clusters of correct aggregation. Likewise, selective attestation will consume more than $O(m)$ bandwidth in such homogeneous situations. For instance with $S = 100\,\text{m}$, selective attestation would consume more bandwidth for any aggregate geographically larger than 1.6 km, as discussed in Section 7.5. Especially in homogeneous traffic situations, such as a traffic jam, such aggregate sizes are likely. In other cases, it is still beneficial to use clustering whenever homogeneity permits. The

Recall that clusters with size $\geq \tau$ will remove prior proofs during hierarchical aggregation, whereas smaller clusters only forward existing proofs.

reason is that clustering is able to perform an explicit agreement between cluster members and create a proof that directly reflects the number of agreeing vehicles. Selective attestation, on the other hand, selects proofs based on geographical distribution. Because vehicles participating in the aggregation do not explicitly agree to the aggregate value but only provide implicit agreement by signing their atomic observations, the semantic proof created by clustering is more likely to detect attackers. The same bandwidth use argument can be made for the multi-signature-based mechanism. Here, all vehicles contribute multi-signatures in the second phase. Therefore, the overhead correlates not only with the geographical size of aggregates but also with the number of aggregate contributors.

Therefore, we argue that the clustering mechanism should be used as the primary security mechanism whenever the average traffic homogeneity $\mathcal{H} \geq \tau$. To amount for short-term effects, we implement the mechanism selection such that mechanism selection is only changed to clustering when the threshold has been reached for a minimum amount of time, and that the selection is only changed back when the threshold has not been reached for a minimum amount of time.

The redundancy mechanism (Chapter 10) can be used orthogonally to the other security mechanisms and independent of traffic homogeneity. The mechanisms goal is to secure the message forwarding path independent of the aggregate values whereas the other mechanisms protect the aggregate values themselves. Therefore, the redundancy mechanism provides an additional line of defense. Even if an attacker is able to alter aggregates and create partial proof data for their correctness, it is still likely that the attacker only controls a fraction of the forwarding paths through the network. Therefore, the results of the redundancy mechanism can be used in parallel to the other mechanisms' results and merged, as discussed in Section 11.2.2, to increase attack detection rates.

11.3.3 *Mechanism combination*

To ensure smooth operation of the underlying aggregation mechanism and information dissemination, it is necessary to gradually combine different security mechanisms. Such combination is feasible, because the security mechanisms' underlying aggregation protocols use compatible data structures for aggregate representation. Hence, aggregates created by different mechanisms can be reused when mechanism selection changes. What differs between mechanisms, however, is the way in which the Decision, Fusion, and Dissemination components are implemented, as well as the way in which proof data is created.

Now suppose a subset of vehicles A has selected to use the clustering mechanism, whereas another subset B uses selective attestation or multisignatures, as shown in Figure 11.2. This situation can exist either because vehicles B have not switched mechanisms yet, or because they are part of a different traffic situation. When vehicles in A receive information from B, they can add it to their world model. The cluster head of A can further decide whether to merge the informa-

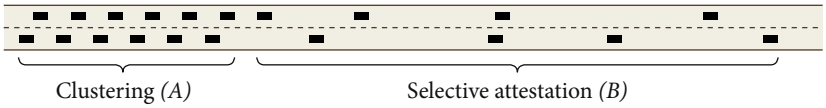


Figure 11.2: Example road with different mechanisms selected.

tion with its own aggregate and disseminate the result. Likewise, when vehicles in B receive aggregates from A's cluster head, they can treat the received aggregates like other aggregates created by vehicles in B. Even if the Fusion components use different algorithms, the results will be compatible as long as both mechanisms use the same data structure.

Proofs from one mechanism, however, cannot simply be merged with proofs from other mechanisms once they are created. Our framework accommodates this by allowing partial proofs on subsets of the aggregates' spatiotemporal area, as shown in Equation (11.1). When the mechanism selection changes, existing proofs will stay attached to aggregates and will be further disseminated. New proofs will be created for newly merged information and all partial proofs will be identified with the region they apply to and the mechanism that created them.

The aggregation lifecycle begins with bootstrapping using atomic observation and continues with an aggregation, forwarding, and possibly further aggregation phase, as described in Section 3.6.4.

We distinguish two variants of proof data combination depending on the aggregate lifecycle. Most likely, the change of mechanisms is due to a change in the underlying traffic situation. Therefore, it is likely that aggregates will change to the forwarding phase after a mechanism change. In this phase, the aggregate is not modified, and the attached proof data does not need to be updated either. If the aggregate is selected for further aggregation, the new mechanism's proof data will be added to the existing proof data without modifying the existing proof data.

Selective attestation → clustering

Let A_1 be an aggregate with an attached proof $\rho_1^{(\mathcal{L}_1,\mathcal{T}_1)}$ that was created using selective attestation. Due to the construction of the selective attestation mechanism, $(\mathcal{L}_1, \mathcal{T}_1)$ will always be equal to the whole aggregate area. Suppose A_1 is now received by a cluster that wants to merge A_1 with its own aggregate A_2 to create $A_1 \ltimes A_2$. The cluster can treat ρ_1 like a proof that was created by another cluster. That is, if the receiving cluster is large enough ($\geq \tau$), it removes ρ_1 and replaces it with a new proof $\rho_2^{(\mathcal{L}_{1\ltimes 2},\mathcal{T}_{1\ltimes 2})}$, as described in Section 9.4.2. If the receiving cluster has fewer than τ members, it forwards ρ_1 unmodified. As soon as an aggregate has passed at least one cluster with τ or more members, it only carries a cluster proof and the mechanism switch is complete.

Clustering → selective attestation

If clustering is disabled and selective attestation enabled, the transition of proof data is more gradual. Selective attestation does not remove old proof data completely. Instead, more and more attestation meta-data is added to an aggregate as its spatiotemporal area grows. Suppose an aggregate A_3 is received, which has been created by a cluster and is accompanied by a cluster agreement proof $\rho_3^{(\mathcal{L}_3,\mathcal{T}_3)}$. The receiving vehicle selects the aggregate for fusion with another aggregate A_4, which carries a selective attestation proof $\rho_4^{(\mathcal{L}_4,\mathcal{T}_4)}$. Fig-

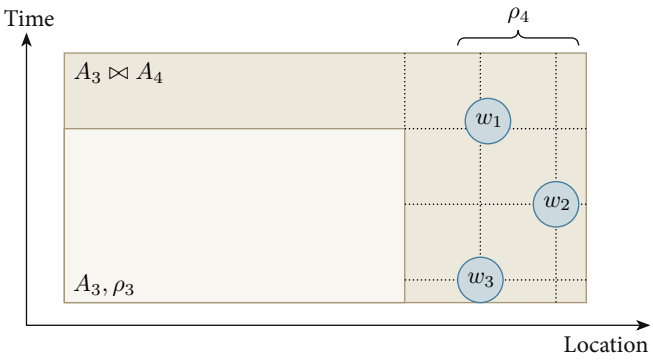


Figure 11.3: Aggregate with prior cluster proof that integrates with selective attestation.

ure 11.3 shows an example. As shown in the figure, the selective attestation grid defined by the security parameter S only applies to the spatiotemporal area covered by A_4. The cluster proof ρ_3 is kept unmodified and concatenated with further meta-data ρ_4. Receiving vehicles will evaluate ρ_3 as if valid selective attestations had been attached to the area $(\mathcal{L}_3, \mathcal{T}_3)$. More specifically, both the selective attestation and the cluster mechanism will each create an opinion on the aggregate. The opinions will be merged as explained in Section 11.2.2. If the aggregate is further merged with other aggregates, ρ_4 is extended and ρ_3 is again left unmodified. The cluster-part of the proof will continue to exist until the aggregate is not disseminated further.

In both directions, selective attestation and clustering can be integrated with a gradual overlap while keeping, further disseminating, and merging existing aggregates. Similar to selective attestation, the multi-signature mechanism can be combined with the clustering mechanism, applying analogous principles during mechanism change. The redundancy mechanism from Chapter 9 can co-exist with all other three mechanisms and provides orthogonal proofs of aggregate integrity.

11.4 ATTACK LIKELIHOOD ADAPTATION

In addition to the adaptation to different traffic situations, security mechanisms should be able to adapt to different attack likelihoods. The security mechanisms we present exhibit configuration parameters that allow to influence the trade-off between bandwidth consumption and resilience against attackers. This influence is a direct one; the more resilient a mechanism is against attackers, the more bandwidth it consumes. Likewise, the less bandwidth a mechanism consumes, the more likely are false positives and false negatives in attack detection if they are not prevented using uncertainty in opinions. Therefore, we propose a mechanism that continuously adapts security mechanisms – more specifically, that adapts the bandwidth that mechanisms consume – based on overall attack likelihood for specific parts of the road network.

Local metric calculation.

To avoid node trust issues, the adaptation metric is calculated locally based on the vehicle's world model. Adaptation influences both creation of a mechanism's proof data, as well as the verification of proof data. Since all vehicles will have similar world model contents once an aggregation mechanism converges, all vehicles will calculate similar attack likelihood and adapt their mechanisms accordingly. Still, adaptation needs to be gradual and verification needs to factor in delays in remote vehicles' adaptation process.

These adaptation requirements can be integrated into our mechanism combination framework, which already uses subjective logic to express confidence in aggregates as continuous values, which adapt to different degrees of confidence. In the following, we derive a metric for attack likelihood from the aggregate-based confidence ratings (Section 11.4.1). We then show how the metric can be continuously updated and adapted without side-effects such as self-amplification (Section 11.4.2). Finally, we show how security mechanisms can be configured based on the metric (Section 11.4.3).

11.4.1 *Attack likelihood metric*

Based on the confidence in aggregate correctness, we define a metric for the likelihood of attacks in certain areas in the road network. Intuitively, the attack likelihood in a given geographical area is inversely proportional to the confidence in the correctness of aggregates within the same area. Let

$$\mathcal{W} = \{(A_1, \omega_1), \ldots, (A_n, \omega_n)\} \tag{11.19}$$

be the world model of a vehicle where each aggregate A_i is annotated with an opinion ω_i as explained in Section 11.2. For attack likelihood, we are only interested in the geographic locations of information, which are given by the locators $\mathcal{L}_i$. To derive attack likelihood from confidence, we use the subjective logic complement operator [94].

DEFINITION 16 (Opinion complement, denoted by $\neg$) Let $\omega = (b, d, u, a)$ be an opinion. Then its complement $\neg\omega$ is defined as

$$\neg\omega : \begin{cases} \neg b = d, \\ \neg d = b, \\ \neg u = u, \\ \neg a = 1 - a. \end{cases} \tag{11.20}$$

Given the locators opinion complements, we have a list

$$\{(\mathcal{L}_1, \neg\omega_1), \ldots, (\mathcal{L}_n, \neg\omega_n),\} \tag{11.21}$$

which represents the likelihood for attacks in certain road network regions. However, the locations in this list may overlap, because the underlying aggregates may overlap. For instance, several aggregates about the same region may exist in the world model, and they may be associated with different confidences.

Next, we split the overlapping locators such that the result is overlap free. For instance, consider two overlapping locators $\mathcal{L}_1 = [x_1, y_1)$ and $\mathcal{L}_2 = [x_2, y_2)$. Without loss of generality, we assume here that locators are overlapping one-dimensional intervals; that is, $x_1, y_1, x_2, y_2 \in \mathbb{R}$ and $y_1 > x_2$. Then the overlapping $\mathcal{L}_1$ and $\mathcal{L}_2$ can be replaced by three new non-overlapping locators

$$\mathcal{L}'_1 = [x_1, x_2), \tag{11.22}$$
$$\mathcal{L}'_2 = [x_2, y_1), \tag{11.23}$$
$$\mathcal{L}'_3 = [y_1, y_2). \tag{11.24}$$

Similarly, locators represented by other data structures than one-dimensional intervals can be made overlap-free. For the overlap part – $\mathcal{L}'_2$ in the example – two opinions about attack likelihood will exist as a result of the process. To merge these opinions, we use the co-multiplication operator of subjective logic.

DEFINITION 17 (Opinion co-multiplication, denoted by $\vee$) Let ω_1 and ω_2 be opinions on the same geographical area. Then their co-multiplied opinion $\omega_1 \vee \omega_2$ is defined as [96]:

$$\omega_1 \vee \omega_2 : \begin{cases} b = b_1 + b_2 - b_1 b_2, \\ d = d_1 d_2 + \frac{(1-a_2)a_1 d_1 u_2 + a_2(1-a_1)u_1 d_2}{a_1 + a_2 - a_1 a_2}, \\ u = u_1 u_2 + \frac{a_2 d_1 u_2 + a_1 u_1 d_2}{a_1 + a_2 - a_1 a_2}, \\ a = a_1 + a_2 - a_1 a_2. \end{cases} \tag{11.25}$$

Note that the co-multiplication of two opinions is equivalent to the multiplication with the roles of belief and disbelief switched.

We use co-multiplication, because the likelihood of an attack should be defined by the opinion representing the highest attack likelihood. This requirement is implemented by the co-multiplication, as indicated by the addition of beliefs $b_1 + b_2$. Intuitively, the co-multiplication corresponds to the logical OR in boolean logic.

The result is an overlap-free list of locators with corresponding merged opinion that represent the attack likelihood for the indicated geographical areas:

$$\{(\mathcal{L}_1, \omega_1), \ldots, (\mathcal{L}_m, \omega_m).\} \tag{11.26}$$

This list is independent of different aggregates and the security mechanisms that contributed the original opinions. Instead, it is a representation of the road network that is annotated with attack likelihood in different geographical areas. We can use this representation to configure different security mechanisms. In particular, the original confidence outputs from one security mechanism can be used to adapt parameters of other security mechanisms. Thus, we leverage the combination of different security mechanisms, which can cross-configure their respective adaptation parameters.

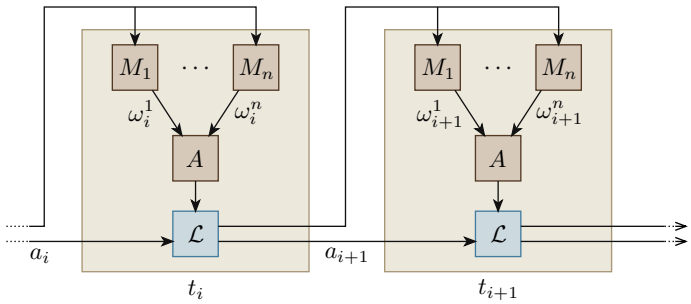


Figure 11.4: Example of temporal adaptation from time $t_i \rightarrow t_{i+1}$.

11.4.2 *Continuous operation*

The metric discussed in Section 11.4.1 represents the attack likelihood for a specific instance in time. However, VANETs are dynamic systems where new information is continuously received and evaluated. Therefore, we need to define how updates are handled and how the attack likelihood metric is updated. Moreover, the adaptation of mechanisms may introduce circular dependencies. Security mechanisms are adapted using attack likelihood. That adaptation may lead to different resulting opinions, which are in turn the basis for the attack likelihood metric. The metric then adapts the mechanisms, which may lead to self-amplification if the adaptation process is not implemented correctly.

Discrete adaptation process.

Therefore, we define a discrete adaptation process. All opinions on aggregates that are created in time period t_i only influence adaptations that are applied in the following time period t_{i+1}. In addition, we feed a summarized version of t_i's attack likelihood into the attack likelihood calculation of t_{i+1}. Figure 11.4 shows an overview of the process. To increase readability, we exemplify the process showing a single aggregate A and its locator $\mathcal{L}$.

In time period t_i, a number of security mechanisms $M_1, \ldots, M_n$ evaluate the aggregate A and express their confidence as belief, disbelief, and uncertainty. The result is translated to attack likelihood as explained in Section 11.4.1. In addition to the explicit belief, disbelief, and uncertainty from the current time period, we use subjective logic's atomicity – also called base rate – to accommodate attack likelihood from previous time periods. In subjective logic, atomicity represents the *a priori* likelihood of an event [219]. In our case, it represents the likelihood of an attack based on historic information and without any opinions created in the current time period. Initially, we set $a_0 = 0.5$ to indicate total uncertainty about attacks.

Calculating atomicity.

Once all mechanisms evaluated A in the current time period t_i, the time period ends. The atomicity a_i is now used in combination with b_i, d_i, and u_i to calculate the expected attack likelihood as discussed in Equation (11.4):

$$E_i = b_i + a_i \cdot u_i. \tag{11.27}$$

In the next time period, t_{i+1}, the expected attack likelihood E_i is used to reconfigure the security mechanisms and adapt their parameters. For higher E_i values, the mechanisms will use more overhead to provide higher resilience. In addition, E_i is used as the *a priori* attack likelihood in the next time period; we set

$$a_{i+1} := E_i. \tag{11.28}$$

By doing so, we use the expected value of the current time period as a bias of opinions in the next time period. As indicated by Equation (11.27), the atomicity influences the result depending on the uncertainty. If mechanisms are uncertain about attack likelihood in the following time period, the next expected value will be influenced by the old value. However, if the mechanism adaptation leads to clearer, more dogmatic ratings of the security mechanisms, the next expected value will be dominated by the mechanism output. This implementation of feedback between time periods prevents self-amplification of attack likelihood.

11.4.3 *Mechanism configuration*

At first, each enabled security mechanism in our framework starts with a default configuration of parameters. Then after each time period, the mechanism is reconfigured using the current expected attack likelihood. The specific implementation of the adaptation is mechanism specific. We discussed specific adaptivity parameters of each security mechanism in Sections 7.7, 8.7, 9.7, and 10.8. Note that the adaptation concerns two parts of the mechanisms.

GENERATION Depending on the current configuration, a mechanism may *attach* more or less proof data to outgoing aggregates.

VALIDATION Depending on the current configuration, a mechanism may *expect* more or less proof data to have been attached by other vehicles. In other words, the same amount of proof data may lead to different confidence values depending on current configuration.

As a result, the adaptation of a security mechanism should be synchronized between different vehicles. Yet, we choose not to implement explicit synchronization, because exchange of synchronization messages may open new attack vectors and adds additional overhead. Instead, each vehicle evaluates the current attack likelihood based on its world model content and the enabled security mechanisms. Given the same initial default configuration and similar world model content, the configuration should at least be *loosely* synchronized.

However, synchronization will not be perfect. Some vehicles may receive aggregates later than other vehicles, which results in later adaptation of attack likelihood. Also, vehicles may join roads on junctions or onramps. These vehicles will not share a similar world model content with other vehicles. Finally, vehicles may not be able to increase proof data on aggregates they only forward. The validation of aggregates should handle these delays gracefully. For

instance, a delayed adaptation may result in lower confidence values for some aggregates, but it should not ignore these aggregates completely. After a certain period of time, the new vehicles will share a common view with existing vehicles, and their world models will converge. By using subjective logic opinions as generalization mechanism output, our framework enables mechanisms to implement such gradual adaptation processes.

EXAMPLE The selective attestation mechanism (Chapter 7) uses a security parameter S, which defines a grid on an aggregate's area. For each grid intersection, one signed atomic observation is kept as attestation meta-data. These signed values together prove the aggregate's correctness. The smaller S, the more bandwidth is used, and the more resilient the security mechanism is against attacks. The value of S can be chosen inversely proportional to the current attack likelihood. If S is adapted during an aggregate's lifetime, a receiving vehicle may expect more atomic observations than are present in the aggregate. The mapping to subjective logic opinions can express this mismatch in a gradual decrease of belief in the aggregate's correctness.

11.4.4 *Summary*

Our mechanism for attack likelihood adaptation leverages the subjective-logic-based framework for aggregate confidence calculation to implement adaptation of different security mechanisms. By deriving a combined expected attack likelihood, the adaptation is easy to implement, because it is based on a single parameter. Using subjective logic atomicity as feedback mechanism allows to take into consideration historic attack likelihood values while preventing self-amplification of adaptation.

11.5 EVALUATION

We separately evaluate our framework's two main concepts. In Section 11.5.1, we assess how different mechanisms can be used for different traffic situations and whether this combination helps to lower bandwidth use while maintaining resilience. Section 11.5.2 evaluates whether the attack likelihood adaptation we proposed in Section 11.4 improves attack detection.

11.5.1 *Traffic context adaptation*

To evaluate the framework's traffic context adaptation, we select two security mechanisms. Namely, we combine selective attestation (Chapter 7) and clustering (Chapter 9), as discussed in Section 11.3.2. The assumption is that selective attestation will incur less overhead in scenarios with low vehicle density, whereas clustering will perform better in scenarios with high vehicle density.

The main goal of traffic context adaptation is to lower the bandwidth overhead compared to using either selective attestation or clustering in all situations.

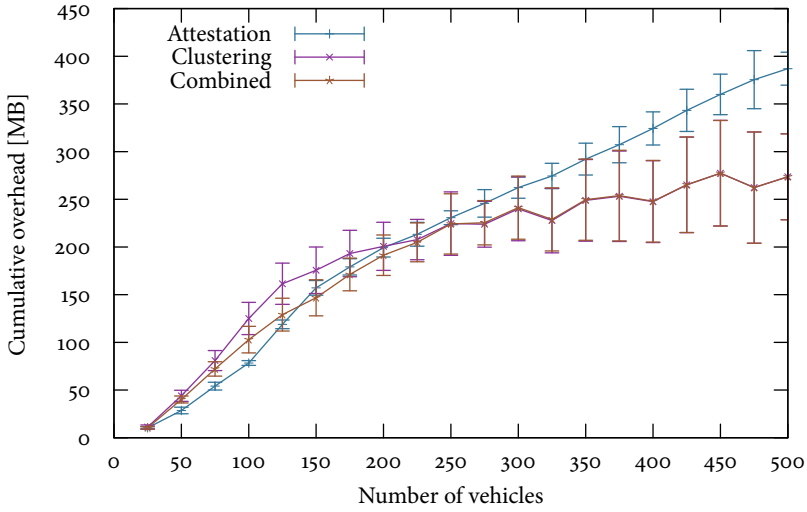


Figure 11.5: Cumulative security overhead comparison for attestation, clustering, and a combined approach.

Since we already evaluated the security properties of the individual mechanisms in Sections 7.4 and 9.5, we therefore focus the combined evaluation on the cumulative security overhead. The cumulative security overhead is calculated as the total overhead generated by all vehicles throughout the whole simulation, measured in megabytes. The total overhead sums up all signatures, public keys, and certificates.

Simulation setup.

The selective attestation mechanism is implemented as described in Chapter 7. The clustering mechanism's intra-cluster communication is implemented as described in Chapter 9. To maximize comparability, the same Decision component is used by the selective attestation mechanism and to determine inter-cluster aggregation in the clustering mechanism. We define a threshold of 30 vehicles within 1-hop communication range to determine handover between attestation and clustering. Selective attestation is used for lower vehicle densities, and clustering is used for higher vehicle densities. We run our simulations on a 5 km stretch of highway with 3 lanes. The highway is blocked by a traffic jam at the right end. Therefore, vehicles will move freely at first before they queue in the developing traffic jam. We vary the vehicle density between 25 and 500 vehicles. For each vehicle density, we simulated the overhead of exclusively using either selective attestation or clustering. In addition, we calculated the combined overhead when using the traffic context adaptation strategy by adding the overhead of each mechanism, assuming it is only used in certain parts of the simulation that are defined by the traffic homogeneity metric.

Simulation results for traffic adaptation.

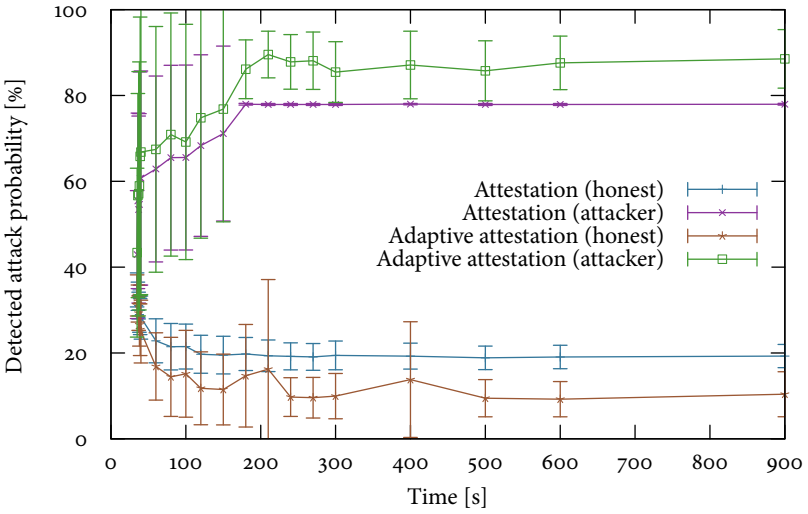
Figure 11.5 shows the simulation results. For vehicle densities between 25 and 200 vehicles, the selective attestation mechanism's overhead is lower. The

reason is that the average cluster size is small in these simulations. Small cluster sizes incur extra overhead, because only large clusters are considered trustworthy by the clustering mechanism. Smaller clusters forward proof data from other clusters, whereas large clusters, which are considered trustworthy, only attach their own proof data. For vehicle densities higher than 200 vehicles, clustering performs better. Here, the average cluster size is larger. Also, clustering benefits from the more efficient communication using cluster heads and gateway vehicles when vehicle density is high and clusters are stable. Moreover, selective attestation attaches proof data with a size linear in the geographic region spanned by aggregates. The overhead due to clustering, however, is constant in the aggregates' geographic region if all clusters are large enough, as discussed in Section 9.4.2. The combined mechanism, which adds the threshold overheads of both mechanisms, is almost equal to the minimum overhead of both mechanisms' total overhead. Especially for higher vehicle densities, the combined overhead is equivalent to the clustering overhead, because clustering is used exclusively as security mechanism. For vehicle densities between 25 and 200 vehicles, the combined overhead is close to but not equal to the selective attestation overhead. The reason is that the threshold criterion is only a heuristic for the best mechanism. Simulation results show that the combined overhead is not optimal but still, on average, close to the better mechanism even for lower vehicle densities. We conclude that the simple traffic density metric defined in Section 11.3.1 offers a good trade-off, because it requires no extra overhead in addition to the aggregation mechanisms and can be calculated locally by each vehicle.

11.5.2 *Mechanism combination and attack likelihood adaptation*

The goal of both mechanism combination (Section 11.2) and attack likelihood adaptation (Section 11.4) is to improve detection of attacks. Therefore, we evaluate both approaches in a combined simulation. As an additional benefit, the adaptation of mechanisms can lower their bandwidth use, which we evaluate as well. To evaluate these goals, we choose a combination of selective attestation (Chapter 7) and the redundancy-based detection (Chapter 9). As argued in Section 11.3.2, the redundancy-based mechanism can run in parallel to other mechanisms to improve attack detection. This improvement is implemented by our mechanism combination approach. Selective attestation can be combined with the redundancy-based approach in low traffic density scenarios. We expect that both mechanisms will complement each other by detecting different kinds of attacks. The redundancy mechanism is expected to work well in situations with homogeneous velocities. In situations where velocity differs in the same location or just changes because of a developing traffic jam, the redundancy mechanism may return false positives. Therefore, we set $\Psi = 0.5$ for the redundancy mechanism. The attestation mechanism depends on cryptographic signatures, and is, therefore, assigned a higher certainty of $\Psi = 0.8$.

(a) 1 attacker



(b) 5 attackers

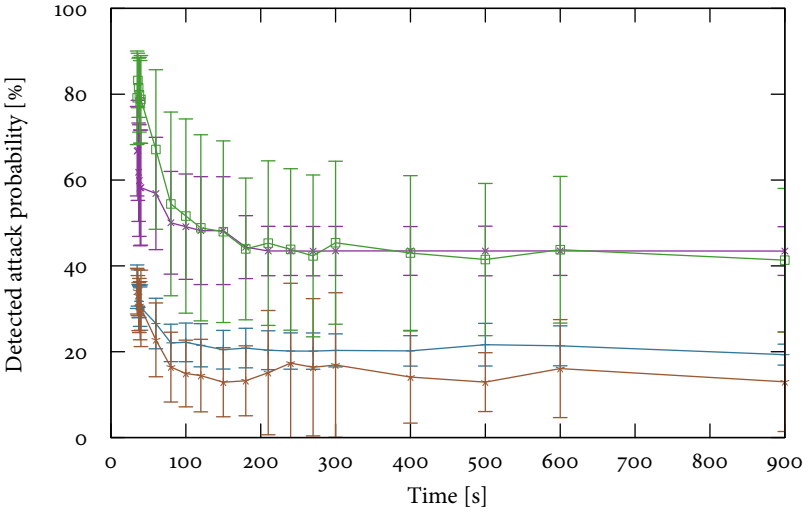


Figure 11.6: Calculated attack probability using the selective attestation mechanism alone.

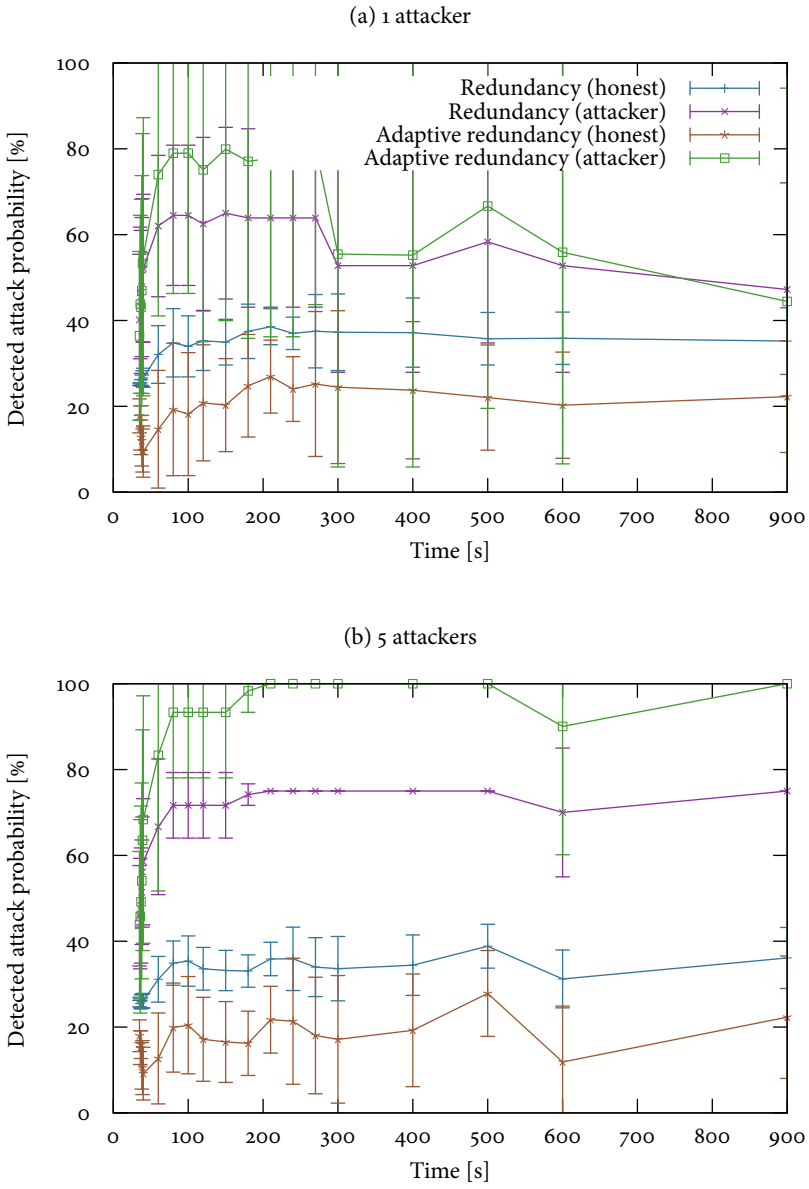


Figure 11.7: Calculated Attack probability using the redundancy mechanism alone.

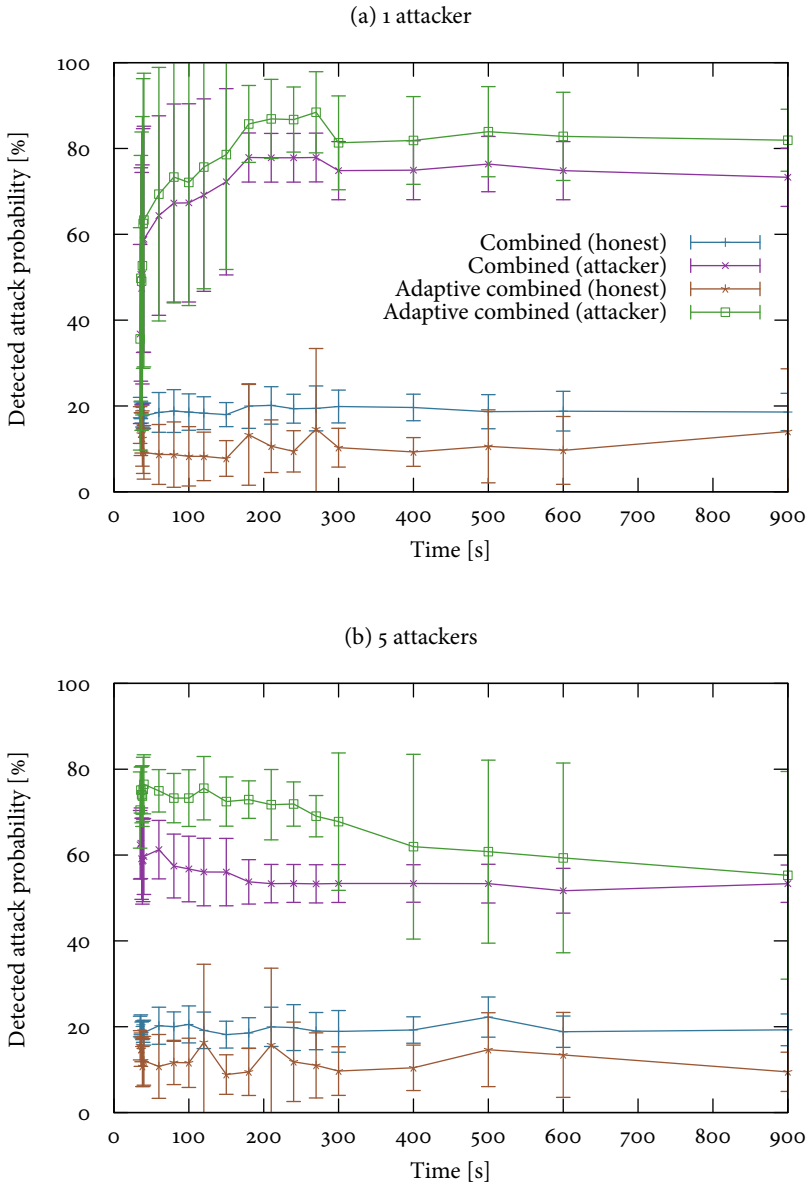


Figure 11.8: Calculated attack probability using a combination of selective attestation and redundancy.

To evaluate the benefits of mechanism combination, we simulate a 8000 m stretch of highway with 3 lanes. Traffic is flowing freely; the vehicle density is 100 vehicles. We test both a single attacker and 5 colluding attackers, which mount a FAKE attack by claiming a traffic jam in the interval 3500–5500 m. To measure the detection rate, we use the average calculated attack probability for the attacked area and the remaining stretch of road. Both attack probabilities are derived by combining all opinions on the respective parts of the world model and calculating their expected value, as discussed in Section 11.2. The selective attestation mechanism is mapped to opinions as explained in Section 11.2.1. The redundancy mechanism detects whether multiple velocities are reported for the same geographic region via node-disjoint forwarding paths. The more the reported velocities differ, the higher derived opinion's belief. Ideally, the calculated attack probability should be significantly larger than 50 % for the attacked region and significantly lower for the remaining region. In addition, the security overhead is calculated in the same way as in Section 11.5.1.

Each mechanism alone, as well as the combination of both mechanisms, is simulated in a static and in an adaptive version. The static version evaluates each instance of the vehicles' world models in isolation. The adaptive versions use the continuous operation mechanism described in Section 11.4.2 to continuously derive atomicity from previous evaluations and feed that into future opinions. Moreover, the selective attestation's S parameter is adapted in the adaptive version. We start with $S = 300\,\text{m}$, meaning 1 signature is added for each 300 m stretch of road covered by an aggregate. For regions with attack probability lower than 30 percent, we set $S = 600\,\text{m}$, and when attack probability exceeds 60 percent, we set $S = 150\,\text{m}$.

Recall that S determines the amount of signatures that are added for a geographic region; higher S values lead to larger overhead and better detection.

Figures 11.6 to 11.8 show the simulation results for attack detection. Each graph shows the simulation time on the x-axis. We use time as x-axis to show how the adaptation influences attack detection over time. The y-axis shows the attack probability for a randomly selected node, which is averaged over 10 random vehicle placements. In each graph, we plot 4 lines. First, we show the detected attack probability of the static mechanism for the non-attacked region. Second, we show the static attack probability for the attacked part of the road. The third and fourth lines show the adaptive attack probabilities for both the non-attacked region and the attacked region, respectively.

Figure 11.6 shows the results for the attestation mechanism alone. In both the 1 attacker setting and the 5 attackers setting, the attack probability is correctly calculated for honest vehicles. Results stabilize at 20% for the static mechanism. The adaptive version stabilizes at 10%, which indicates a benefit of the adaptation. Namely, low attack probabilities are emphasized over time; that is, historic ratings bias current ratings correctly. The results for the attacked road regions differ between the 1 attacker and 5 attacker setting. In the 1 attacker setting, the calculated attack probability is high in the static setting and even higher in the adaptive setting. But for 5 attackers, the static attestation mechanism rates the attack probability considerably lower. The reason is that 5 attackers can produce more valid signatures, which reflects in less certain opinions. The adaptation

mechanism corrects this behavior, because the S parameter is adapted when signs for an attack are detected. Once S is adapted, the attackers cannot attach more signatures to their fake aggregates, whereas the honest nodes can select more signatures to be kept for correct aggregates. Therefore, the adaptive attestation version performs considerably better for higher amounts of attackers.

The results of the redundancy mechanism versions, depicted in Figure 11.7, show a different behavior. Overall, the redundancy mechanism performs better in the 5 attackers setting. More specifically, the results for the adaptive and static version are consistent for honest vehicles. As expected the detected attack probability is higher than for the selective attestation mechanism. The reason is that heterogeneous traffic may introduce false positives. However, all honest aggregates are rated on average lower than 50%. The results again improve when adaptation is used. The detection of attacker aggregates is – unlike the attestation case – better in the 5 attackers setting. In the 1 attacker setting, the redundancy mechanism introduces false negatives for longer simulation times. That is, attacker aggregates are rated below 50% for both the static and the adaptive version. The reason for this drop in performance is that the redundancy mechanism benefits from the additional forwarding paths introduced by more attackers. The more aggregates are received about the same region, the more consistently can conflicts be detected.

Evaluation summary.

We summarize that the attestation mechanism performs especially well in the 1 attacker setting, whereas the redundancy mechanism shows better results in the 5 attacker setting. Figure 11.8 shows how the combination of both mechanisms eliminates the drawbacks of the separate mechanisms. The graphs show that the 1 attacker setting, as is to be expected, results in a clearer distinction between attacker aggregates and honest information. However, the 5 attackers setting also clearly distinguishes attacker aggregates from honest information. The results for the combined mechanisms, therefore, are the only results that are consistent for both simulated attacker amounts.

In all evaluation scenarios, the adaptation of mechanisms has improved detection results. Figure 11.9 shows an additional benefit of using mechanism adaptation. Because the selective attestation mechanism only uses a large number of attached signatures in regions where attacks are likely, the cumulative bandwidth usage decreases when adaptation is used. We conclude that the combination of mechanisms, when used with mechanism adaptation, both improves detection probability and lowers bandwidth usage.

11.6 SUMMARY

We have presented a comprehensive framework to integrate different security mechanisms from Chapters 7 to 10. By introducing a subjective-logic-based reasoning approach, we create the foundation for a flexible combination of mechanisms in a range of different scenarios. The generic reasoning enables mechanism combination in two different dimensions. First, we adapt the selection of mechanisms based on the current traffic situation. The goal of this

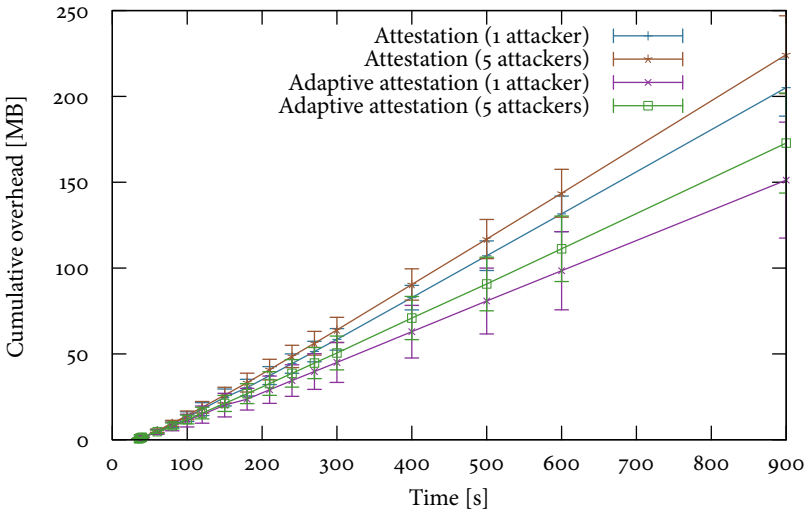


Figure 11.9: Security overhead with and without adaptation.

adaptation is mainly to reduce the bandwidth use while maintaining proper protection against attackers. Second, we adapt mechanisms to achieve consistent attack detection for a variety of scenarios. This combination both improves attack detection, and it further reduces bandwidth use.

To instantiate our framework, we discussed how our security mechanisms perform in different scenarios and how they are used in different components of the framework. We verify our proposals with comprehensive simulations. The simulation results show that our framework is able to improve attack detection and bandwidth usage at the same time. Thereby, the framework eliminates some of the trade-offs between bandwidth usage and aggregation efficiency we discussed in Chapters 7 to 10.

Beyond the security mechanisms presented in Chapters 7 to 10, a subset of the framework's components may be applicable to other use cases beyond in-network aggregation. For instance, the generic representation and combination of mechanism output may be used in other situations where multiple mechanisms are used to jointly evaluate information in a world model. Also, mechanism adaptation may be applied when suitable adaptation parameters can be defined. Also, adaptation to different contexts may be applicable to other use cases. However, it depends on the use case whether a suitable metric for mechanism selection can be defined.

IV

CONCLUSIONS

CONCLUSIONS AND FUTURE WORK

12

12.1 OVERVIEW

In this thesis, we have designed, evaluated, and integrated resilient in-network aggregation protocols for VANETs to facilitate long-range traffic efficiency applications. In-network aggregation arguably is a necessity to implement scalable protocols for traffic efficiency applications [152]. At the same time, semantic summarization of information, as it is applied by in-network aggregation, complicates information integrity protection. In particular, adding resilience by protecting information integrity often comes at the cost of additional overhead. Therefore, we have posed the main research question:

> How can the resilience of an in-network information aggregation process and integrity of aggregated information be ensured while maintaining the bandwidth benefits introduced by aggregation in VANETs?

12.2 DISCUSSION

In-network aggregation modeling.

As our first step towards building resilient in-network aggregation mechanisms, we have established an understanding of how aggregation protocols work and how they disseminate information within the network. To this end, we have contributed a generic architecture and information flow model (Contribution a). Our architecture model identifies mandatory components of in-network aggregation mechanisms (Question 1). We have characterized information flow by introducing an abstract aggregation operator, identifying distinct aggregation lifecycle phases, and differentiating patterns within the information flow (Question 2).

While related work often uses components of the architecture model implicitly [e. g., 114, 126], to the best of our knowledge, no existing work provides similar generic models for aggregation mechanism architectures. A main result of our information flow model is the observation of redundant communication paths, which has inspired the design of our redundancy-based security mechanism.

Security analysis for in-network aggregation.

Based on the analysis of in-network aggregation itself, we have identified assets of aggregation mechanisms (Question 3) and discussed threats to these assets. Our categorization of different security approaches (Contribution b) provides the basis for a structured research methodology (Question 4).

Mechanism design and evaluation.

To analyze the benefits and limitations of different approaches to achieve resilient aggregation (Question 5), we have proposed four specific security mechanisms (Contribution c). Each proposed mechanism has been evaluated in terms of its resilience and its bandwidth overhead (Contribution d).

Cryptography-based mechanisms.

The first security mechanism we present (Chapter 7) is based on cryptographic message protection in combination with data consistency checks. The main contribution of this mechanism is its applicability to very dynamic aggregation protocols, such as DYN. In contrast to related work [146], no fixed segmentation nor any kind of interaction between vehicles are required.

However, the bandwidth overhead of the mechanism grows linear in the aggregate area for a fixed security parameters. While we have proposed adaptation mechanisms for the security parameters, the bandwidth overhead may remain problematic for situations where aggregates need to span large geographic regions.

Therefore, we have proposed an alternative cryptographic security mechanism based on sketches and multi-signatures in Chapter 8. The mechanism improves on existing resilient mechanisms proposed by Garofalakis et al. [72], Han et al. [78], and Hsiao et al. [82], which also use sketches as underlying data structure. Namely, our mechanism does not require central authorities (in contrast to [78]) nor does it require prior agreement on aggregated values (in contrast to [72, 82]).

While our sketches-based mechanism provides security for very dynamic aggregation settings, we have observed that it introduces a considerable security overhead, because several cryptographic signatures and key material are required to protect the integrity of the underlying FM sketches. These drawbacks are partially eliminated by the mechanism's second phase where multi-signatures are applied to reduce bandwidth overhead when information is disseminated in farther away regions. But the overhead in the mechanism's first phase in combination with the requirement for multiple phases may render the mechanism inefficient for aggregation of information about large geographic regions.

Clustering-based interactive mechanism.

As both our cryptography-based security mechanisms suffer from high bandwidth use in large areas or dense traffic scenarios, our third mechanism proposal, a cluster-based security mechanism (Chapter 9), specifically addresses these scenarios. We have shown that aggregation decisions, such as velocity similarity can be applied to create stable clusters. The main advantage of our clustering-based resilience mechanism is that its overhead is constant rather than linear in the size of the aggregates' geographic regions, because we have designed a bandwidth-efficient proof of cluster size that is used during inter-cluster communication. The mechanism thereby improves over a proposal of Raya and Hubaux [146], where clusters are built using fixed geographic regions, and where hierarchical aggregation is not explicitly addressed. The mechanism's main drawback is that its applicability to sparse traffic situations may be limited, because it requires a minimum cluster size to be resilient against attackers.

Redundancy-based mechanism.

In Chapter 10, we have proposed a mechanism that is based on communication redundancy in combination with data-consistency mechanisms. Our redundancy-based mechanism is orthogonal especially to the proposals in Chapters 7 and 8. A main contribution is the introduction of secure path lists, which make our redundancy-based mechanism applicable to the dynamic aggrega-

tion setting. In contrast to existing work, as well as to our mechanisms in Chapters 7 and 8, the mechanism's overhead is limited to the size of the secure path lists, which makes it scalable. In sparse traffic situations, however, the mechanism may only be able to detect inconsistencies rather then determine and filter the correct information.

Fundamental limits.

All four mechanisms rely on variations of the assumption that information that is reported by a large number of participants is likely correct. To determine that a large number of participants report about an event, their identities need to be distinguished. Our mechanisms implement different approaches to reduce the security overhead while maintaining resilience against attackers. But evaluation results indicate that there may be an intrinsic trade-off between overhead and resilience, partially addressing Question 6.

Mechanism combination and adaptivity.

To further lower bandwidth overhead while maintaining resilience against attacks, we have proposed a mechanism combination and adaptation framework (Contribution e). Specifically, we have shown how cryptography-based security mechanisms can be combined with interactive mechanisms to achieve protection against attacks in both sparse traffic situations, as well as high density traffic, while maintaining tolerable overhead. As an orthogonal mode of adaptation, we have introduced a generic mechanism to adapt each mechanism's overhead based on the current attack likelihood. This adaptation strategy allows to further improve bandwidth usage while maintaining resilience. In contrast to other generic frameworks, such as [75], our framework is flexible enough to be applied to in-network aggregation. Due to its generic design, the framework may well be applied to other use cases where insider attackers need to be detected. Our evaluation results have shown that our framework achieves protection against a wider range of attacks while lowering security overhead at the same time (Question 7).

Addressing our main research question.

We conclude that *resilience of in-network aggregation against insider attackers can be achieved* by employing a number of different mechanisms, which we explored in a structured way based on our security paradigms taxonomy. However, each mechanism introduces considerable overhead and is efficiently applicable only to specific traffic situations. Therefore, to *maintain the bandwidth benefits introduced by aggregation,* a combination of different mechanisms and their adaptation to different attack likelihoods is required. We have proposed a generic framework and demonstrated using simulations that such combination and adaptation is feasible.

12.3 OUTLOOK

In this work, we have demonstrated that resilient in-network aggregation is achievable and that the combination and adaptation of different mechanisms allows to achieve resilience while maintaining bandwidth efficiency. We see three areas where future work could contribute to our proposals: evaluation in experimental deployments, formal models, and generalization to other use cases and research domains.

Testbed evaluation.

We have evaluated our mechanism proposals using network simulations. Although simulation environments offer a controlled environment, the evaluation of our results in real deployments would underline the feasibility of our contributions. As we target traffic efficiency applications, a large number of vehicles would be required for evaluation in deployments. Although field operational trials for VANETs are underway as of 2015, there is no testbed available that would provide the required number of vehicles to test scalability of in-network aggregation mechanisms. Moreover, we consider testing of resilience in real deployments an open challenge. A fully operational deployment of traffic efficiency applications is required to acquire knowledge about attacks on real systems. Traces of such attacks would help to validate our results in more realistic settings. But even if such attack traces would be available, it is a challenge to replay them to test resilience mechanisms without endangering actual drivers on the roads.

Formal proofs.

As second direction, we consider the formal proof of lower bounds on bandwidth use for certain resilience levels. Similar proofs have been published by Bhaskar et al. [28] for specific aggregation mechanisms in WSNs. However, their results cannot be directly applied to the different network setting in VANETs. The results of our simulations indicate that similar bounds may apply to VANETs, but a formal proof of these bounds still requires further investigations.

Application to other domains.

Finally, we see the potential for applying our results in other domains besides VANETs. Since the advent of small form factor devices with sufficient processing power and networking capabilities, more and more applications diverge from traditional centralized networks. Examples are wireless sensor networks or urban sensing [20, 107]. As such networks are composed of large numbers of devices, distributing processing and fusion of information throughout the network is often a necessity for achieving sufficient scalability. The need for information aggregation will create security challenges that are very related to the ones discussed in this thesis, and we assume that our solutions can be adapted to such scenarios.

Furthermore, our framework for combination and adaptivity of mechanisms (Chapter 11) has the potential to be applied to other domains than in-network aggregation, such as misbehavior detection in other mobile ad hoc networks. Likewise, our redundancy-based security approach (Chapter 10) may be applicable to other mobile routing scenarios with redundant message dissemination. Therefore, while the solutions we present in this thesis are tailored towards vehicular networks, the underlying concepts can serve as templates to ensure resilience in a wide range of networks that follow the same trend.

ACKNOWLEDGEMENTS

Writing this thesis is probably the most challenging, intense, and enjoyable endeavor I have undertaken so far. Many people have been a part of the journey by providing guidance, comments, support, and distractions. I am immensely grateful to all of you.

In late 2008, I joined the research group of Frank Kargl, and he has done a tremendous job in being my promotor ever since. Frank gave me the freedom to explore any scientific direction I found interesting, and he provided guidance when I needed it. *Danke,* Frank!

Under Frank's guidance, I worked in three research groups during my time as a doctoral candidate: the Institute of Media Informatics and Institute of Distributed Systems at Ulm University, Germany, and the Distributed and Embedded Systems Group at the University of Twente, Netherlands. Working for different groups gave me the opportunity to meet many inspiring scientists.

Michael Weber, chair of the Institute of Media Informatics, continues to inspire me with his perseveringly subtle way to lead his institute. Pieter Hartel, chair of the Distributed and Embedded Systems group, has always fascinated me with his uncompromising pursuit of science. *Danke,* Michael and *bedankt,* Pieter, for allowing me to work in your groups. Geert Heijenk, whom I met at the University of Twente, has since become my co-promoter. *Bedankt,* Geert, for your guidance during and after my time in Twente and for your constructive and uplifting comments on my thesis.

I also wish to thank the members of my graduation committee that I have not addressed so far – Peter Apers, Aiko Pras, Björn Scheuermann, Franz Hauck, and Elmar Schoch – for their service and insightful comments.

Florian Schaub has been and continues to be an invaluable friend for over twenty years. Florian has always been available for discussion and distraction, and he inspired me with his persistently positive attitude towards life. *Danke,* Florian, for your friendship, your guidance, and for your meticulously thorough comments that helped to improve my thesis.

Many colleagues shared parts of my journey towards the doctorate degree. Each of you has influenced my scientific and non-scientific outlook on life in one way or another. *Danke,* Elmar, for teaching me the 101 of the science business. *Danke,* Thorsten, for telling me things about writing a doctoral thesis that made so much sense years after you said them. *Danke,* Christoph, for teaching me to relax when stress is uncalled for. *Hvala,* Dina, for always being positively honest. *Merci,* Jonathan, for sharing your cheerful spirit with me.

Danke, Boto, Zhendong, Bastian, Jonas, Frank, Felix, and Marc, for sharing my journey at the Institute of Media Informatics. Thank you, Arjan, Saeed, Luan, Trajce, Begül, Richard, Qiang, Andre, Ayşe, Wolter, Elmer, Eelco, Svetla, Emmanuele, Michael, and Damiano, for being part of my time at the Distributed and Embedded Security Group. *Und danke,* Christian, Steffen, Rens, Henning,

Thomas, Stephan, and Benjamin, for the time we shared at the Institute of Distributed Systems. I also wish to thank Claudia, Nienke, Bertine, Ida, Suse, Marion, Anja, and Kathrin for organizing all the big and little things during my university employment.

Thank you, Maximilian, Andreas, Sung-Eun, Benjamin, Christian, Stefan, Guiseppe, Naim, Rens, Julian, and Mihail, for your great projects, Diplom theses, and master theses that I had the pleasure to supervise.

At least as important as my scientific fellows are my friends that helped me to cope with the ups and downs of scientific life. Thank you, Moritz, Laura, Jonas, Matthias, Nina, Michael, Nadine, Bastian, Sarah, Rob, Sara, Cindy, Christina, Thorsten, Pam, Christian, Ashleigh, Susi, Andrea, Tamer, and many more.

Schließlich danke ich meiner Familie und insbesondere meinen Eltern Helga und Heinrich. Ihr habt mich zu dem gemacht der ich bin und mir ermöglicht alle meine Träume zu verwirklichen. Danke!

Thank you all,
Stefan

Ulm, March 2015

BIBLIOGRAPHY

PEER-REVIEWED PUBLICATIONS BY THE AUTHOR

[1] S. Dietzel, R. W. van der Heijden, H. Decke, and F. Kargl. "A flexible, subjective logic-based framework for misbehavior detection in V2V networks." In: *IEEE 15th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. June 2014, pp. 1–6. DOI: 10.1109/WoWMoM.2014.6918989 (cit. on pp. 10, 191).

[2] S. Dietzel, J. Petit, F. Kargl, and B. Scheuermann. "In-Network Aggregation for Vehicular Ad Hoc Networks." In: *IEEE Communications Surveys and Tutorials* 16.4 (2014), pp. 1909–1932. ISSN: 1553-877X. DOI: 10.1109/COMST.2014.2320091 (cit. on pp. 6–8, 21, 23, 27, 29, 53, 83, 95).

[3] S. Dietzel, B. Bako, E. Schoch, and F. Kargl. "A Fuzzy Logic Based Approach for Structure-Free Aggregation in Vehicular ad-hoc Networks." In: *Sixth ACM international workshop on VehiculAr InterNETworking (VANET)*. ACM Press, 2009, pp. 79–88. ISBN: 9781605587370. DOI: 10.1145/1614269.1614283 (cit. on pp. 7, 21, 23, 25, 31, 34, 39–41, 50, 70, 150).

[4] S. Dietzel, M. Balanici, and F. Kargl. "Short paper: Towards data-similarity-based clustering for inter-vehicle communication." In: *IEEE Vehicular Networking Conference (VNC)*. Dec. 2013, pp. 238–241. DOI: 10.1109/VNC.2013.6737622 (cit. on pp. 10, 23, 75, 149).

[5] S. Dietzel, J. Gürtler, R. W. van der Heijden, and F. Kargl. "Redundancy-based Statistical Analysis for Insider Attack Detection in VANET Aggregation Schemes." In: *IEEE Vehicular Networking Conference (VNC)*. Paderborn, Germany, Dec. 2014. DOI: 10.1109/VNC.2014.7013332 (cit. on pp. 10, 169).

[6] S. Dietzel, F. Kargl, G. Heijenk, and F. Schaub. "Modeling in-Network Aggregation in VANETs." In: *IEEE Communications Magazine* 49.11 (Nov. 2011), pp. 142–148. DOI: 10.1109/MCOM.2011.6069721 (cit. on pp. 7, 21).

[7] S. Dietzel, F. Kargl, G. Heijenk, and F. Schaub. "On the Potential of Generic Modeling for VANET Data Aggregation Protocols." In: *2nd IEEE Vehicular Networking Conference (VNC)*. IEEE, 2010, pp. 78–85. DOI: 10.1109/VNC.2010.5698256 (cit. on pp. 7, 21).

[8] S. Dietzel, M. Kost, F. Schaub, and F. Kargl. "CANE: A Controlled Application Environment for privacy protection in ITS." In: *12th International Conference on ITS Telecommunications (ITST)*. Nov. 2012, pp. 71–76. DOI: 10.1109/ITST.2012.6458663 (cit. on p. 79).

[9] S. Dietzel, A. Peter, and F. Kargl. "Secure Cluster-based In-network Information Aggregation for Vehicular Networks." In: *IEEE 81st Vehicular Technology Conference (VTC)*. To appear: IEEE, 2015 (cit. on pp. 10, 149).

[10] S. Dietzel, J. Petit, G. Heijenk, and F. Kargl. "Graph-Based Metrics for Insider Attack Detection in VANET Multihop Data Dissemination Protocols." In: *IEEE Transactions on Vehicular Technology* 62.4 (2013), pp. 1505–1518. DOI: 10.1109/TVT.2012.2236117 (cit. on pp. 8, 46, 78, 169, 178).

[11] S. Dietzel, J. Petit, F. Kargl, and G. Heijenk. "Analyzing Dissemination Redundancy to Achieve Data Consistency in VANETs." In: *Ninth ACM international workshop on VehiculAr InterNETworking (VANET)*. ACM, June 2012, pp. 131–134. DOI: 10.1145/2307888.2307914 (cit. on pp. 8, 46, 78).

[12] S. Dietzel, E. Schoch, B. Bako, and F. Kargl. "A Structure-Free Aggregation Framework for Vehicular ad hoc Networks." In: *6th International Workshop on Intelligent Transportation (WIT)*. 2009, pp. 61–66 (cit. on pp. 21, 50).

[13] S. Dietzel, E. Schoch, B. Könings, M. Weber, et al. "Resilient Secure Aggregation for Vehicular Networks." In: *IEEE Network* 24.1 (2010), pp. 26–31. DOI: 10.1109/MNET.2010.5395780 (cit. on pp. 10, 89, 109).

[14] R. W. van der Heijden, S. Dietzel, and F. Kargl. "SeDyA: secure dynamic aggregation in VANETs." In: *ACM conference on Security and privacy in wireless and mobile networks (WiSec)*. WiSec. Budapest, Hungary: ACM, 2013, pp. 131–142. ISBN: 978-1-4503-1998-0. DOI: 10.1145/2462096.2462119 (cit. on pp. 10, 27, 75, 131).

[15] F. Kargl, F. Schaub, and S. Dietzel. "Mandatory Enforcement of Privacy Policies using Trusted Computing Principles." In: *Intelligent Information Privacy Management Symposium*. AAAI, 2010, pp. 104–109 (cit. on p. 79).

[16] B. Könings, F. Schaub, F. Kargl, and S. Dietzel. "Channel switch and quiet attack: New DoS attacks exploiting the 802.11 standard." In: *IEEE 34th Conference on Local Computer Networks (LCN)*. 2009, pp. 14–21. DOI: 10.1109/LCN.2009.5355149 (cit. on p. 60).

[17] M. Kost, B. Wiedersheim, S. Dietzel, F. Schaub, et al. "WiSec 2011 demo: PRECIOSA PeRA – practical enforcement of privacy policies in intelligent transportation systems." In: *ACM SIGMOBILE Mobile Computing and Communications Review* 15.3 (Nov. 2011), pp. 37–38. DOI: 10.1145/2073290.2073298 (cit. on p. 79).

[18] E. Schoch, B. Bako, S. Dietzel, and F. Kargl. "Dependable and secure geocast in vehicular networks." In: *Seventh ACM international workshop on VehiculAr InterNETworking (VANET)*. ACM Press, Sept. 2010, pp. 61–68. DOI: 10.1145/1860058.1860068 (cit. on pp. 77, 137).

PEER-REVIEWED REFERENCES

[19] N. Ahmed, T. Natarajan, and K. R. Rao. "Discrete Cosine Transform." In: *Computers, IEEE Transactions on* C-23.1 (Jan. 1974), pp. 90–93. ISSN: 0018-9340 (cit. on p. 32).

[20] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. "Wireless sensor networks: a survey." In: *Computer Networks* 38.4 (2002), pp. 393–422. ISSN: 1389-1286. DOI: http://dx.doi.org/10.1016/S1389-1286(01)00302-4 (cit. on pp. 4, 28, 83, 222).

[21] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. "A survey on sensor networks." In: *IEEE Communications Magazine* 40.8 (Aug. 2002), pp. 102–114. ISSN: 0163-6804. DOI: 10.1109/MCOM.2002.1024422 (cit. on p. 84).

[22] H. Alzaid, E. Foo, and J. G. Nieto. "Secure Data Aggregation in Wireless Sensor Network: A Survey." In: *Proceedings of the Sixth Australasian Conference on Information Security - Volume 81*. AISC '08. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2008, pp. 93–105. ISBN: 978-1-920682-62-0 (cit. on pp. 80, 84).

[23] D. Antolino Rivas, J. M. Barceĺo-Ordinas, M. Guerrero Zapata, and J. D. Morillo-Pozo. "Security on VANETs: Privacy, Misbehaving Nodes, False Information and Secure Data Aggregation." In: *Journal of Network and Computer Applications* 34.6 (Nov. 2011), pp. 1942–1955 (cit. on p. 88).

[24] B. Bako, F. Kargl, E. Schoch, and M. Weber. "Advanced Adaptive Gossiping Using 2-Hop Neighborhood Information." In: *IEEE Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008*. Nov. 2008, pp. 1–6. DOI: 10.1109/GLOCOM.2008.ECP.1045 (cit. on pp. 16, 19, 137, 178, 186).

[25] N. Banerjee, M. D. Corner, D. Towsley, and B. N. Levine. "Relays, base stations, and meshes: enhancing mobile networks with infrastructure." In: *Proceedings of the 14th ACM international conference on Mobile computing and networking*. San Francisco, California, USA: ACM, 2008, pp. 81–91. ISBN: 978-1-60558-096-8 (cit. on p. 16).

[26] C. Barba, M. Mateos, P. Soto, A. Mezher, et al. "Smart city for VANETs using warning messages, traffic statistics and intelligent traffic lights." In: *2012 IEEE Intelligent Vehicles Symposium (IV)*. June 2012, pp. 902–907. DOI: 10.1109/IVS.2012.6232229 (cit. on p. 13).

[27] J. L. Bentley. "Multidimensional binary search trees used for associative searching." In: *Commun. ACM* 18.9 (Sept. 1975), pp. 509–517. ISSN: 0001-0782 (cit. on p. 31).

[28] R. Bhaskar, R. Jaiswal, and S. Telang. "Congestion lower bounds for secure in-network aggregation." In: *Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 2012, pp. 197–204 (cit. on pp. 64, 85, 86, 91, 189, 222).

[29] N. Bismeyer, S. Mauthofer, K. M. Bayarou, and F. Kargl. "Assessment of node trustworthiness in VANETs using data plausibility checks with particle filters." In: *Proceedings of the 2012 IEEE Vehicular Networking Conference (VNC)*. IEEE, Nov. 2012, pp. 78–85. ISBN: 978-1-4673-4996-3, 978-1-4673-4995-6, 978-1-4673-4994-9. DOI: 10.1109/VNC.2012.6407448 (cit. on p. 87).

[30] N. Bissmeyer, C. Stresing, and K. Bayarou. "Intrusion detection in VANETs through verification of vehicle movement data." In: *2010 IEEE Vehicular Networking Conference (VNC)*. Dec. 2010, pp. 166–173. DOI: 10.1109/VNC.2010.5698232 (cit. on p. 87).

[31] A. Boldyreva. "Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme." In: *Public Key Cryptography*. Vol. 2567. 2003, pp. 31–46 (cit. on pp. 74, 90, 131).

[32] A. Boldyreva, C. Gentry, A. O'Neill, and D. H. Yum. "New Multiparty Signature Schemes for Network Routing Applications." In: *ACM Trans. Inf. Syst. Secur.* 12.1 (Oct. 2008), 3:1–3:39. ISSN: 1094-9224. DOI: 10.1145/1410234.1410237 (cit. on p. 73).

[33] D. Boneh and D. M. Freeman. "Homomorphic Signatures for Polynomial Functions." In: *Advances in Cryptology – EUROCRYPT 2011*. Ed. by K. G. Paterson. Lecture Notes in Computer Science 6632. Springer Berlin Heidelberg, Jan. 1, 2011, pp. 149–168. ISBN: 978-3-642-20464-7, 978-3-642-20465-4 (cit. on p. 75).

[34] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps." In: *Advances in Cryptology — EUROCRYPT 2003*. Ed. by E. Biham. Lecture Notes in Computer Science 2656. Springer Berlin Heidelberg, Jan. 1, 2003, pp. 416–432. ISBN: 978-3-540-14039-9, 978-3-540-39200-2 (cit. on pp. 74, 90, 120).

[35] D. Boneh, B. Lynn, and H. Shacham. "Short Signatures from the Weil Pairing." In: *Advances in Cryptology — ASIACRYPT 2001*. Ed. by C. Boyd. Lecture Notes in Computer Science 2248. Springer Berlin Heidelberg, Jan. 1, 2001, pp. 514–532. ISBN: 978-3-540-42987-6, 978-3-540-45682-7 (cit. on pp. 74, 86, 90, 122).

[36] L. Borges, F. Velez, and A. Lebres. "Survey on the Characterization and Classification of Wireless Sensor Networks Applications." In: *IEEE Communications Surveys Tutorials* 16.4 (2014), pp. 1860–1890. ISSN: 1553-877X. DOI: 10.1109/COMST.2014.2320073 (cit. on p. 14).

[37] A. Buchenscheit, F. Schaub, F. Kargl, and M. Weber. "A VANET-based emergency vehicle warning system." In: *2009 IEEE Vehicular Networking Conference (VNC)*. Oct. 2009, pp. 1–8. DOI: 10.1109/VNC.2009.5416384 (cit. on p. 16).

[38] L. Buttyán, P. Schaffer, and I. Vajda. "RANBAR: RANSAC-based Resilient Aggregation in Sensor Networks." In: *Proceedings of the Fourth ACM Workshop on Security of Ad Hoc and Sensor Networks*. SASN '06. New York, NY, USA: ACM, 2006, pp. 83–90. ISBN: 1-59593-554-1. DOI: 10.1145/1180345.1180356 (cit. on p. 104).

[39] L. Buttyàn, P. Schaffer, and I. Vajda. "Resilient Aggregations: Statistical Approach." In: *Sensor Networks and Configuration*. Ed. by N. P. Mahalik. Springer Berlin Heidelberg, Jan. 1, 2007, pp. 211–236. ISBN: 978-3-540-37364-3, 978-3-540-37366-7 (cit. on p. 104).

[40] N. Cai and T. Chan. "Theory of Secure Network Coding." In: *Proceedings of the IEEE* 99.3 (Mar. 2011), pp. 421–437. ISSN: 0018-9219. DOI: 10.1109/JPROC.2010.2094592 (cit. on p. 75).

[41] G. Calandriello, P. Papadimitratos, J.-P. Hubaux, and A. Lioy. "On the Performance of Secure Vehicular Communication Systems." In: *IEEE Transactions on Dependable and Secure Computing* 8.6 (Nov. 2011), pp. 898–912. ISSN: 1545-5971. DOI: 10.1109/TDSC.2010.58 (cit. on p. 59).

[42] M. Caliskan, D. Graupner, and M. Mauve. "Decentralized Discovery of Free Parking Places." In: *VANET '06: Proceedings of the 3rd acm International Workshop on Vehicular ad hoc Networks*. ACM, 2006, pp. 30–39 (cit. on pp. 13, 23, 26, 29).

[43] J. Camenisch, S. Hohenberger, and M. Ø. Pedersen. "Batch Verification of Short Signatures." In: *Advances in Cryptology - EUROCRYPT 2007*. Ed. by M. Naor. Lecture Notes in Computer Science 4515. Springer Berlin Heidelberg, Jan. 1, 2007, pp. 246–263. ISBN: 978-3-540-72539-8, 978-3-540-72540-4 (cit. on p. 86).

[44] C. Castelluccia, E. Mykletun, and G. Tsudik. "Efficient aggregation of encrypted data in wireless sensor networks." In: *The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2005. MobiQuitous 2005*. July 2005, pp. 109–117. DOI: 10.1109/MOBIQUITOUS.2005.25 (cit. on p. 85).

[45] G.-Y. Chang, J.-P. Sheu, and C.-Y. Chung. "Zooming: a Zoom-Based Approach for Parking Space Availability in VANET." In: *Vehicular Technology Conference (vtc 2010-Spring), 2010 Ieee 71st*. IEEE, 2010, pp. 1–5 (cit. on pp. 26, 32, 39).

[46] W. Chen. "VANETs-Based Real-Time Traffic Data Dissemination." In: *Wireless Communications, Networking and Information Security (Wcnis), 2010 IEEE International Conference on*. IEEE, 2010, pp. 468–472 (cit. on p. 34).

[47] J. C. Choon and J. H. Cheon. "An Identity-Based Signature from Gap Diffie-Hellman Groups." In: *Public Key Cryptography — PKC 2003*. Ed. by Y. G. Desmedt. Lecture Notes in Computer Science 2567. Springer Berlin Heidelberg, Jan. 1, 2002, pp. 18–30. ISBN: 978-3-540-00324-3, 978-3-540-36288-3 (cit. on p. 75).

[48] F. Cuckov and M. Song. "Geocast-Driven Structureless Information Dissemination Scheme for Vehicular Ad Hoc Networks." In: *2010 IEEE Fifth International Conference on Networking, Architecture and Storage (NAS)*. July 2010, pp. 325–332. DOI: 10.1109/NAS.2010.51 (cit. on p. 29).

[49] F. Cuckov and M. Song. "Geocast-Driven Structureless Information Dissemination Scheme for Vehicular ad hoc Networks." In: *Networking, Architecture and Storage (nas), 2010 Ieee Fifth International Conference on*. IEEE, 2010, pp. 325–332 (cit. on p. 34).

[50] M. L. Das. "A Key Escrow-Free Identity-Based Signature Scheme without using Secure Channel." In: *Cryptologia* 35.1 (2010), pp. 58–72 (cit. on p. 131).

[51] S. Dashtinezhad, T. Nadeem, B. Dorohonceanu, C. Borcea, et al. "TrafficView: a driver assistant device for traffic monitoring based on car-to-car communication." In: *Vehicular Technology Conference, 2004. vtc 2004-Spring. 2004 Ieee 59th*. Vol. 5. May 2004, pp. 2946–2950 (cit. on pp. 30, 39, 40, 150).

[52] B. Defude, T. Delot, S. Ilarri, J.-L. Zechinelli, et al. "Data Aggregation in VANETs: the Vespa Approach." In: *Proceedings of the 5th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*. Dublin, Ireland: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, 13:1–13:6. ISBN: 978-963-9799-27-1 (cit. on p. 23).

[53] M. v. Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. "Fully Homomorphic Encryption over the Integers." In: *Advances in Cryptology – EUROCRYPT 2010*. Ed. by H. Gilbert. Lecture Notes in Computer Science 6110. Springer Berlin Heidelberg, Jan. 1, 2010, pp. 24–43. ISBN: 978-3-642-13189-9, 978-3-642-13190-5 (cit. on p. 75).

[54] M. Ding, X. Cheng, and G. Xue. "Aggregation tree construction in sensor networks." In: *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*. Vol. 4. IEEE, 2003, pp. 2168–2172 (cit. on p. 28).

[55] D. Dolev and A. C. Yao. "On the security of public key protocols." In: *IEEE Transactions on Information Theory* 29.2 (Mar. 1983), pp. 198–208. ISSN: 0018-9448. DOI: 10.1109/TIT.1983.1056650 (cit. on p. 61).

[56] F. Dötzer. "Privacy Issues in Vehicular Ad Hoc Networks." In: *Privacy Enhancing Technologies*. Ed. by G. Danezis and D. Martin. Lecture Notes in Computer Science 3856. Springer Berlin Heidelberg, Jan. 1, 2006, pp. 197–209. ISBN: 978-3-540-34745-3, 978-3-540-34746-0 (cit. on p. 59).

[57] J. Douceur. "the Sybil Attack." In: *Iptps '01: First International Workshop on Peer-to-Peer Systems*. Springer, 2002, pp. 251–260 (cit. on pp. 5, 59).

[58] C. Dwork and M. Naor. "Pricing via Processing or Combatting Junk Mail." In: *Advances in Cryptology — CRYPTO' 92*. Ed. by E. F. Brickell. Lecture Notes in Computer Science 740. Springer Berlin Heidelberg, Jan. 1, 1993, pp. 139–147. ISBN: 978-3-540-57340-1, 978-3-540-48071-6 (cit. on p. 87).

[59] J. Edmonds and R. M. Karp. "Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems." In: *J. ACM* 19.2 (Apr. 1972), pp. 248–264. ISSN: 0004-5411. DOI: 10.1145/321694.321699 (cit. on p. 177).

[60] M. van Eenennaam and G. Heijenk. "Providing over-the-Horizon Awareness to Driver Support Systems." In: *Proceedings of 4th Ieee Workshop on Vehicle to Vehicle Communications (V2vcom)*. 2008 (cit. on pp. 31, 32, 40).

[61] S. Eichler. "Performance Evaluation of the IEEE 802.11p WAVE Communication Standard." In: *2007 IEEE 66th Vehicular Technology Conference*. IEEE, 2007, pp. 2199–2203 (cit. on p. 18).

[62] S. Eichler, C. Schroth, T. Kosch, and M. Strassberger. "Strategies for Context-Adaptive Message Dissemination in Vehicular Ad Hoc Networks." In: *Mobile and Ubiquitous Systems: Networking Services, 2006 Third Annual International Conference on*. July 2006, pp. 1–9 (cit. on pp. 34, 41).

[63] Y.-C. Fan and A. Chen. "Efficient and robust sensor data aggregation using linear counting sketches." In: *IEEE International Symposium on Parallel and Distributed Processing, 2008. IPDPS 2008*. Apr. 2008, pp. 1–12. DOI: 10.1109/IPDPS.2008.4536265 (cit. on p. 133).

[64] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi. "In-network aggregation techniques for wireless sensor networks: a survey." In: *IEEE Wireless Communications* 14.2 (Apr. 2007), pp. 70–87. ISSN: 1536-1284. DOI: 10.1109/MWC.2007.358967 (cit. on pp. 22, 28).

[65] M. Feiri, J. Petit, and F. Kargl. "Congestion-based Certificate Omission in VANETs." In: *Proceedings of the Ninth ACM International Workshop on Vehicular Inter-networking, Systems, and Applications*. VANET '12. New York, NY, USA: ACM, 2012, pp. 135–138. ISBN: 978-1-4503-1317-9. DOI: 10.1145/2307888.2307915 (cit. on p. 59).

[66] C. Feng, Z. Li, S. Jiang, and R. Zhang. "Data Aggregation and Routing Guidance with QoS Guarantee in VANETs." In: *International Journal of Distributed Sensor Networks* 2014 (July 3, 2014), e262437. ISSN: 1550-1329. DOI: 10.1155/2014/262437 (cit. on p. 23).

[67] M. Ferreira, R. Fernandes, H. Conceição, W. Viriyasitavat, et al. "Self-organized Traffic Control." In: *Proceedings of the Seventh ACM International Workshop on VehiculAr InterNETworking*. VANET '10. New York, NY, USA: ACM, 2010, pp. 85–90. ISBN: 978-1-4503-0145-9. DOI: 10.1145/1860058.1860077 (cit. on p. 13).

[68] R. Finkel and J. Bentley. "Quad trees: a data structure for retrieval on composite keys." In: *Acta Informatica* 4.1 (1974). ISSN: 0001-5903. DOI: 10.1007/BF00288933 (cit. on p. 29).

[69] P. Flajolet, É. Fusy, O. Gandouet, and F. Meunier. "HyperLogLog: the analysis of a near-optimal cardinality estimation algorithm." In: *DMTCS Proceedings* 1 (Jan. 20, 2008). ISSN: 1365-8050 (cit. on pp. 103, 156–158).

[70] P. Flajolet and G. N. Martin. "Probabilistic Counting Algorithms for Data Base Applications." In: *Journal of Computer and System Sciences* 31.2 (Oct. 1985), pp. 182–209 (cit. on pp. 27, 133).

[71] A. A. Freitas. "A Survey of Evolutionary Algorithms for Data Mining and Knowledge Discovery." In: *Advances in Evolutionary Computing*. Ed. by D. A. Ghosh and P. D. S. Tsutsui. Natural Computing Series. Springer Berlin Heidelberg, Jan. 1, 2003, pp. 819–845. ISBN: 978-3-642-62386-8, 978-3-642-18965-4 (cit. on p. 27).

[72] M. Garofalakis, J. M. Hellerstein, and P. Maniatis. "Proof Sketches: Verifiable in-Network Aggregation." In: *Data Engineering, 2007. Icde 2007. Ieee 23rd International Conference on*. IEEE, 2007, pp. 996–1005 (cit. on pp. 27, 86, 90, 102, 103, 135, 220).

[73] C. Gentry and Z. Ramzan. "Identity-Based Aggregate Signatures." In: *Public Key Cryptography - PKC 2006*. Ed. by M. Yung, Y. Dodis, A. Kiayias, and T. Malkin. Lecture Notes in Computer Science 3958. Springer Berlin Heidelberg, Jan. 1, 2006, pp. 257–273. ISBN: 978-3-540-33851-2, 978-3-540-33852-9 (cit. on pp. 73, 120, 122, 123, 136).

[74] O. Goldreich, S. Micali, and A. Wigderson. "Proofs That Yield Nothing but Their Validity or All Languages in NP Have Zero-knowledge Proof Systems." In: *J. ACM* 38.3 (July 1991), pp. 690–728. ISSN: 0004-5411. DOI: 10.1145/116825.116852 (cit. on p. 76).

[75] P. Golle, D. Greene, and J. Staddon. "Detecting and correcting malicious data in VANETs." In: *Proceedings of the first ACM workshop on Vehicular ad hoc networks - VANET '04*. ACM Press, 2004, p. 29. ISBN: 1581139225 (cit. on pp. 6, 221).

[76] V. Gradinescu, C. Gorgorin, R. Diaconescu, V. Cristea, et al. "Adaptive Traffic Lights Using Car-to-Car Communication." In: *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*. Apr. 2007, pp. 21–25. DOI: 10.1109/VETECS.2007.17 (cit. on p. 13).

[77] Y. Gunter, B. Wiegel, and H. Grossmann. "Cluster-based Medium Access Scheme for VANETs." In: *IEEE Intelligent Transportation Systems Conference, 2007. ITSC 2007*. Sept. 2007, pp. 343–348. DOI: 10.1109/ITSC.2007.4357651 (cit. on p. 149).

[78] Q. Han, S. Du, D. Ren, and H. Zhu. "SAS: a Secure Data Aggregation Scheme in Vehicular Sensing Networks." In: *Proceedings of the 2010 Ieee International Conference on Communications (icc)*. IEEE, 2010, pp. 1–5 (cit. on pp. 27, 90, 91, 102, 103, 135, 138, 220).

[79] H. Hartenstein and K. Laberteaux. "A tutorial survey on vehicular ad hoc networks." In: *IEEE Communications Magazine* 46.6 (June 2008), pp. 164–171. ISSN: 0163-6804. DOI: 10.1109/MCOM.2008.4539481 (cit. on pp. 3, 11).

[80] D. Helbing and M. Treiber. "Gas-Kinetic-Based Traffic Model Explaining Observed Hysteretic Phase Transition." In: *Physical Review Letters* 81.14 (Oct. 1998), pp. 3042–3045. ISSN: 0031-9007, 1079-7114. DOI: 10.1103/PhysRevLett.81.3042 (cit. on p. 78).

[81] S. Heule, M. Nunkesser, and A. Hall. "HyperLogLog in Practice: Algorithmic Engineering of a State of the Art Cardinality Estimation Algorithm." In: *Proceedings of the 16th International Conference on Extending Database Technology*. EDBT '13. New York, NY, USA: ACM, 2013, pp. 683–692. ISBN: 978-1-4503-1597-5. DOI: 10.1145/2452376.2452456 (cit. on pp. 156, 161, 163).

[82] H.-C. Hsiao, A. Studer, R. Dubey, E. Shi, et al. "Efficient and Secure Threshold-Based Event Validation for VANETs." In: *Proceedings of acm Conference on Wireless Network Security (Wisec)*. 2011 (cit. on pp. 5, 23, 87, 102, 156, 158, 220).

[83] K. Ibrahim and M. Weigle. "CASCADE: Cluster-Based Accurate Syntactic Compression of Aggregated Data in VANETs." In: *2008 IEEE Globecom Workshops*. IEEE, Nov. 2008, pp. 1–10. ISBN: 978-1-4244-3061-1 (cit. on pp. 32, 40).

[84] K. Ibrahim and M. Weigle. "Towards an Optimized and Secure Cascade for Data Aggregation in VANETs." In: *Proceedings of the Fifth acm International Workshop on Vehicular Inter-Networking - VANET '08*. ACM Press, 2008. ISBN: 9781605581910 (cit. on p. 89).

[85] K. Ibrahim and M. C. Weigle. "Optimizing Cascade Data Aggregation for VANETs." In: *5th Ieee International Conference on Mobile ad hoc and Sensor Systems*. IEEE, Sept. 2008, pp. 724–729. ISBN: 978-1-4244-2574-7 (cit. on p. 32).

[86] K. Ibrahim, M. Weigle, and G. yan. "Light-Weight Laser-Aided Position Verification for Cascade." In: *Proceedings of the International Conference on Wireless Access in Vehicular Environments (Wave)*. Dec. 2008 (cit. on p. 89).

[87] "IEEE Guide for Wireless Access in Vehicular Environments (WAVE) - Architecture." In: *IEEE Std 1609.0-2013* (2013) (cit. on p. 15).

[88] "IEEE Standard for Message Sets for Vehicle/Roadside Communications." In: *IEEE Std 1455-1999* (Sept. 1999), pp. 1–134. DOI: 10.1109/IEEESTD.1999.90564 (cit. on p. 15).

[89] "IEEE Standard for Wireless Access in Vehicular Environments Security Services for Applications and Management Messages." In: *IEEE Std 1609.2-2013 (Revision of IEEE Std 1609.2-2006)* (Apr. 2013), pp. 1–289. DOI: 10.1109/IEEESTD.2013.6509896 (cit. on p. 57).

[90] A. Jaeger, N. Bißmeyer, H. Stübing, and S. A. Huss. "A Novel Framework for Efficient Mobility Data Verification in Vehicular Ad-hoc Networks." In: *International Journal of Intelligent Transportation Systems Research* 10.1 (Jan. 1, 2012), pp. 11–21. ISSN: 1348-8503, 1868-8659. DOI: 10.1007/s13177-011-0038-9 (cit. on pp. 77, 87).

[91] S. Jaggi, M. Langberg, S. Katti, T. Ho, et al. "Resilient network coding in the presence of Byzantine adversaries." In: *IEEE INFOCOM 2007. 26th IEEE International Conference on Computer Communications*. May 2007, pp. 616–624. DOI: 10.1109/INFCOM.2007.78 (cit. on p. 75).

[92] S. Joerer, F. Dressler, and C. Sommer. "Comparing Apples and Oranges?: Trends in IVC Simulations." In: *Proceedings of the Ninth ACM International Workshop on Vehicular Inter-networking, Systems, and Applications*. VANET '12. Low Wood Bay, Lake District, UK: ACM, 2012, pp. 27–32. ISBN: 978-1-4503-1317-9. DOI: 10.1145/2307888.2307895 (cit. on p. 97).

[93] D. Johnson, A. Menezes, and S. Vanstone. "The Elliptic Curve Digital Signature Algorithm (ECDSA)." In: *International Journal of Information Security* 1.1 (2001), pp. 36–63. ISSN: 1615-5262. DOI: 10.1007/s102070100002 (cit. on p. 58).

[94] A. Jøsang. "A Logic for Uncertain Probabilities." In: *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* 9.3 (June 2001), pp. 279–311. ISSN: 0218-4885 (cit. on pp. 193, 204).

[95] A. Jøsang, S. Marsh, and S. Pope. "Exploring Different Types of Trust Propagation." In: *Trust Management*. Ed. by K. Stølen, W. H. Winsborough, F. Martinelli, and F. Massacci. Lecture Notes in Computer Science 3986. Springer Berlin Heidelberg, Jan. 1, 2006, pp. 179–192. ISBN: 978-3-540-34295-3, 978-3-540-34297-7 (cit. on pp. 195, 196).

[96] A. Jøsang and D. McAnally. "Multiplication and comultiplication of beliefs." In: *International Journal of Approximate Reasoning* 38.1 (Jan. 2005), pp. 19–51. ISSN: 0888-613X. DOI: 10.1016/j.ijar.2004.03.003 (cit. on pp. 196, 197, 205).

[97] J. Al-Karaki and A. Kamal. "Routing techniques in wireless sensor networks: a survey." In: *IEEE Wireless Communications* 11.6 (Dec. 2004), pp. 6–28. ISSN: 1536-1284. DOI: 10.1109/MWC.2004.1368893 (cit. on p. 14).

[98] F. Kargl, P. Papadimitratos, L. Buttyan, M. Müter, et al. "Secure vehicular communication systems: implementation, performance, and research challenges." In: *IEEE Communications Magazine* 46.11 (Nov. 2008), pp. 110–118. ISSN: 0163-6804. DOI: 10.1109/MCOM.2008.4689253 (cit. on p. 57).

[99] M. Killat, F. Schmidt-Eisenlohr, H. Hartenstein, C. Rössel, et al. "Enabling Efficient and Accurate Large-scale Simulations of VANETs for Vehicular Traffic Management." In: *Proceedings of the Fourth ACM International Workshop on Vehicular Ad Hoc Networks*. VANET '07. New York, NY, USA: ACM, 2007, pp. 29–38. ISBN: 978-1-59593-739-1. DOI: 10.1145/1287748.1287754 (cit. on p. 96).

[100] T. H.-J. Kim, A. Studer, R. Dubey, X. Zhang, et al. "VANET Alert Endorsement Using Multi-source Filters." In: *Proceedings of the Seventh ACM International Workshop on VehiculAr InterNETworking*. VANET '10. New York, NY, USA: ACM, 2010, pp. 51–60. ISBN: 978-1-4503-0145-9. DOI: 10.1145/1860058.1860067 (cit. on pp. 87, 92).

[101] D. E. Knuth. "Big Omicron and Big Omega and Big Theta." In: *SIGACT News* 8.2 (Apr. 1976), pp. 18–24. ISSN: 0163-5700. DOI: 10.1145/1008328.1008329 (cit. on pp. 25, 86).

[102] A. N. Kolmogorov. "On tables of random numbers." In: *Theoretical Computer Science* 207.2 (1998), pp. 387–395 (cit. on p. 73).

[103] R. Kumar and M. Dave. "A Framework For Handling Local Broadcast Storm Using Probabilistic Data Aggregation In VANET." In: *Wireless Personal Communications* (2013), pp. 1–27. ISSN: 0929-6212 (cit. on p. 31).

[104] R. Kumar and M. Dave. "Knowledge Based Framework for Data Aggregation in Vehicular Ad Hoc Networks." In: *Computational Intelligence and Information Technology*. Ed. by V. V. Das and N. Thankachan. Vol. 250. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 722–727. ISBN: 978-3-642-25733-9, 978-3-642-25734-6 (cit. on p. 31).

[105] N. Labraoui, M. Gueroui, M. Aliouat, and J. Petit. "Adaptive security level for data aggregation in Wireless Sensor Networks." In: *2010 5th IEEE International Symposium on Wireless Pervasive Computing (ISWPC)*. May 2010, pp. 325–330. DOI: 10.1109/ISWPC.2010.5483752 (cit. on p. 85).

[106] N. Labraoui, M. Gueroui, M. Aliouat, and J. Petit. "Reactive and adaptive monitoring to secure aggregation in wireless sensor networks." In: *Telecommunication Systems* 54.1 (Sept. 1, 2013), pp. 3–17. ISSN: 1018-4864, 1572-9451. DOI: 10.1007/s11235-013-9712-3 (cit. on p. 85).

[107] U. Lee and M. Gerla. "A survey of urban vehicular sensing platforms." In: *Computer Networks* 54.4 (Mar. 19, 2010), pp. 527–544. ISSN: 1389-1286. DOI: 10.1016/j.comnet.2009.07.011 (cit. on p. 222).

[108] T. Leinmuller, R. Schmidt, E. Schoch, A. Held, et al. "Modeling Roadside Attacker Behavior in VANETs." In: *2008 IEEE GLOBECOM Workshops*. Nov. 2008, pp. 1–10. DOI: 10.1109/GLOCOMW.2008.ECP.63 (cit. on p. 5).

[109] T. Leinmüller, C. Maihöfer, E. Schoch, and F. Kargl. "Improved Security in Geographic Ad Hoc Routing Through Autonomous Position Verification." In: *Proceedings of the 3rd International Workshop on Vehicular Ad Hoc Networks*. VANET '06. New York, NY, USA: ACM, 2006, pp. 57–66. ISBN: 1-59593-540-1. DOI: 10.1145/1161064.1161075 (cit. on pp. 77, 87).

[110] F. Li and Y. Wang. "Routing in vehicular ad hoc networks: A survey." In: *IEEE Vehicular Technology Magazine* 2.2 (June 2007), pp. 12–22. ISSN: 1556-6072. DOI: 10.1109/MVT.2007.912927 (cit. on p. 150).

[111] S. Lindsey and C. Raghavendra. "PEGASIS: Power-efficient gathering in sensor information systems." In: *IEEE Aerospace Conference Proceedings, 2002*. Vol. 3. 2002, pp. 1125–1130. DOI: 10.1109/AERO.2002.1035242 (cit. on p. 28).

[112] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, et al. "The 1999 DARPA off-line intrusion detection evaluation." In: *Computer Networks* 34.4 (Oct. 2000), pp. 579–595. ISSN: 1389-1286. DOI: 10.1016/S1389-1286(00)00139-0 (cit. on p. 98).

[113] C. Lochert, B. Scheuermann, and M. Mauve. "A Probabilistic Method for Cooperative Hierarchical Aggregation of Data in VANETs." In: *Elsevier Ad Hoc Networks* 8.5 (2010), pp. 518–530. DOI: 10.1016/j.adhoc.2009.12.008 (cit. on pp. 13, 33, 39).

[114] C. Lochert, B. Scheuermann, and M. Mauve. "Probabilistic Aggregation for Data Dissemination in VANETs." In: *Proceedings of the Fourth acm International Workshop on Vehicular ad hoc Networks - Vanet '07*. ACM Press, 2007, pp. 1–8. ISBN: 9781595937391 (cit. on pp. 13, 23, 25–27, 33, 90, 103, 132, 133, 158, 219).

[115] C. Lochert, B. Scheuermann, C. Wewetzer, A. Luebke, et al. "Data Aggregation and Roadside Unit Placement for a VANET Traffic Information System." In: *Proceedings of the Fifth acm International Workshop on Vehicular Inter-Networking - VANET '08*. ACM Press, 2008, p. 58. ISBN: 9781605581910 (cit. on pp. 16, 23, 24, 26, 30).

[116] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. "TAG: A Tiny AGgregation Service for Ad-hoc Sensor Networks." In: *SIGOPS Oper. Syst. Rev.* 36 (SI Dec. 2002), pp. 131–146. ISSN: 0163-5980. DOI: 10.1145/844128.844142 (cit. on p. 28).

[117] A. Mahimkar and T. Rappaport. "SecureDAV: a secure data aggregation and verification protocol for sensor networks." In: *IEEE Global Telecommunications Conference, 2004. GLOBECOM '04*. Vol. 4. Nov. 2004, 2175–2179 Vol.4. DOI: 10.1109/GLOCOM.2004.1378395 (cit. on p. 85).

[118] T. Mangel and H. Hartenstein. "An analysis of data traffic in cellular networks caused by inter-vehicle communication at intersections." In: *2011 IEEE Intelligent Vehicles Symposium (IV)*. June 2011, pp. 473–478. DOI: 10.1109/IVS.2011.5940495 (cit. on p. 16).

[119] T. Mangel, F. Schweizer, T. Kosch, and H. Hartenstein. "Vehicular safety communication at intersections: Buildings, Non-Line-Of-Sight and representative scenarios." In: *2011 Eighth International Conference on Wireless On-Demand Network Systems and Services (WONS)*. Jan. 2011, pp. 35–41. DOI: 10.1109/WONS.2011.5720197 (cit. on p. 14).

[120] T. Mangel, M. Michl, O. Klemp, and H. Hartenstein. "Real-World Measurements of Non-Line-Of-Sight Reception Quality for 5.9GHz IEEE 802.11p at Intersections." In: *Communication Technologies for Vehicles*. Ed. by T. Strang, A. Festag, A. Vinel, R. Mehmood, et al. Lecture Notes in Computer Science 6596. Springer Berlin Heidelberg, Jan. 1, 2011, pp. 189–202. ISBN: 978-3-642-19785-7, 978-3-642-19786-4 (cit. on p. 14).

[121] M. Milojevic and V. Rakocevic. "Distributed road traffic congestion quantification using cooperative VANETs." In: *Ad Hoc Networking Workshop (MED-HOC-NET), 2014 13th Annual Mediterranean*. June 2014, pp. 203–210. DOI: 10.1109/MedHocNet.2014.6849125 (cit. on p. 23).

[122] S. Mohanty and D. Jena. "Secure Data Aggregation in Vehicular-Adhoc Networks: a Survey." In: *Procedia Technology* 6 (Jan. 2012), pp. 922–929 (cit. on p. 88).

[123] J. M. Molina-Gil, P. Caballero-Gil, C. Herńandez-Goya, and Ć. Caballero-Gil. "Data Aggregation for Information Authentication in VANETs." In: *Information Assurance and Security (ias), 2010 Sixth International Conference on*. 2010, pp. 282–287 (cit. on pp. 89, 92).

[124] J. Molina-Gil, P. Caballero-Gil, and C. Caballero-Gil. "Aggregation and probabilistic verification for data authentication in VANETs." In: *Information Sciences* 262 (Mar. 20, 2014), pp. 172–189. ISSN: 0020-0255. DOI: 10.1016/j.ins.2013.07.036 (cit. on pp. 89, 91, 92, 126).

[125] T. Nadeem, S. Dashtinezhad, C. Liao, and L. Iftode. "TrafficView: a scalable traffic monitoring system." In: *Mobile Data Management, 2004. Proceedings. 2004 Ieee International Conference on*. 2004, pp. 13–26 (cit. on p. 30).

[126] T. Nadeem, S. Dashtinezhad, C. Liao, and L. Iftode. "TrafficView: Traffic Data Dissemination Using car-to-car Communication." In: *ACM SIGMOBILE Mobile Computing and Communications Review* 8.3 (July 2004). ISSN: 15591662 (cit. on pp. 30, 39, 219).

[127] T. Nadeem, P. Shankar, and L. Iftode. "A Comparative Study of Data Dissemination Models for VANETs." In: *3rd Annual International Conference on Mobile and Ubiquitous Systems Mobiquitous*. IEEE, 2006, pp. 1–10. ISBN: 1424404983 (cit. on p. 30).

[128] K. Nagel and M. Schreckenberg. "A cellular automaton model for freeway traffic." In: *Journal de Physique I* 2.12 (Dec. 1992), pp. 2221–2229. ISSN: 1155-4304, 1286-4862. DOI: 10.1051/jp1:1992277 (cit. on p. 78).

[129] E. F. Nakamura, A. A. F. Loureiro, and A. C. Frery. "Information Fusion for Wireless Sensor Networks: Methods, Models, and Classifications." In: *ACM Comput. Surv.* 39.3 (Sept. 2007). ISSN: 0360-0300. DOI: 10.1145/1267070.1267073 (cit. on pp. 22, 28).

[130] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu. "The Broadcast Storm Problem in a Mobile Ad Hoc Network." In: *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*. MobiCom '99. New York, NY, USA: ACM, 1999, pp. 151–162. ISBN: 1-58113-142-9. DOI: 10.1145/313451.313525 (cit. on p. 17).

[131] E. Palomar, J. M. de Fuentes, A. I. González-Tablas, and A. Alcaide. "Hindering false event dissemination in VANETs with proof-of-work mechanisms." In: *Transportation Research Part C: Emerging Technologies* 23 (Aug. 2012), pp. 85–97. ISSN: 0968-090X. DOI: 10.1016/j.trc.2011.08.002 (cit. on p. 87).

[132] P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, et al. "Secure vehicular communication systems: design and architecture." In: *IEEE Communications Magazine* 46.11 (Nov. 2008), pp. 100–109. ISSN: 0163-6804. DOI: 10.1109/MCOM.2008.4689252 (cit. on pp. 57, 67).

[133] H. J. Payne. "FREFLO: A Macroscopic Simulation Model of Freeway Traffic." In: *Transportation Research Record* 722 (1979). ISSN: 0361-1981 (cit. on p. 78).

[134] T. P. Pedersen. "A Threshold Cryptosystem without a Trusted Party." In: *Advances in Cryptology — EUROCRYPT '91*. Ed. by D. W. Davies. Lecture Notes in Computer Science 547. Springer Berlin Heidelberg, Jan. 1, 1991, pp. 522–526. ISBN: 978-3-540-54620-7, 978-3-540-46416-7 (cit. on p. 85).

[135] Y. Peng, Z. Abichar, and J. Chang. "Roadside-Aided Routing (RAR) in Vehicular Networks." In: *IEEE International Conference on Communications, 2006. ICC '06*. Vol. 8. June 2006, pp. 3602–3607. DOI: 10.1109/ICC.2006.255631 (cit. on p. 16).

[136] A. Perrig, R. Canetti, J. Tygar, and D. Song. "The TESLA Broadcast Authentication Protocol." In: *RSA CryptoBytes* 5 (Summer 2002) (cit. on p. 91).

[137] J. Petit, M. Feiri, and F. Kargl. "Spoofed data detection in VANETs using dynamic thresholds." In: *2011 IEEE Vehicular Networking Conference (VNC)*. Nov. 2011, pp. 25–32. DOI: 10.1109/VNC.2011.6117120 (cit. on pp. 5, 86, 87).

[138] J. Petit and Z. Mammeri. "Dynamic consensus for secured vehicular ad hoc networks." In: *2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. Oct. 2011, pp. 1–8. DOI: 10.1109/WiMOB.2011.6085418 (cit. on pp. 70, 86).

[139] J. Petit, F. Schaub, M. Feiri, and F. Kargl. "Pseudonym Schemes in Vehicular Networks: A Survey." In: *IEEE Communications Surveys Tutorials* PP.99 (2014), pp. 1–1. ISSN: 1553-877X. DOI: 10.1109/COMST.2014.2345420 (cit. on p. 58).

[140] J. Petit and Z. Mammeri. "Authentication and consensus overhead in vehicular ad hoc networks." In: *Telecommunication Systems* 52.4 (Apr. 1, 2013), pp. 2699–2712. ISSN: 1018-4864, 1572-9451. DOI: 10.1007/s11235-011-9589-y (cit. on p. 86).

[141] F. Picconi, N. Ravi, M. Gruteser, and L. Iftode. "Probabilistic Validation of Aggregated Data in Vehicular ad-hoc Networks." In: *Proceedings of the 3rd International Workshop on Vehicular ad hoc Networks - VANET '06*. ACM Press, 2006, p. 76. ISBN: 1595935401 (cit. on pp. 32, 67, 76, 88).

[142] B. Przydatek, D. Song, and A. Perrig. "SIA: Secure Information Aggregation in Sensor Networks." In: *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*. SenSys '03. New York, NY, USA: ACM, 2003, pp. 255–265. ISBN: 1-58113-707-9. DOI: 10.1145/958491.958521 (cit. on pp. 75, 84, 85).

[143] Z. Rawshdeh and S. Mahmud. "Toward Strongley Connected Clustering Structure in Vehicular Ad Hoc Networks." In: *Vehicular Technology Conference Fall (VTC 2009-Fall), 2009 IEEE 70th*. Sept. 2009, pp. 1–5. DOI: 10.1109/VETECF.2009.5378799 (cit. on pp. 149, 151).

[144] M. Raya, P. Papadimitratos, V. Gligor, and J.-P. Hubaux. "On Data-Centric Trust Establishment in Ephemeral Ad Hoc Networks." In: *IEEE INFOCOM 2008. The 27th Conference on Computer Communications*. Apr. 2008. DOI: 10.1109/INFOCOM.2008.180 (cit. on pp. 102, 103, 110).

[145] M. Raya, A. Aziz, and J.-P. Hubaux. "Efficient Secure Aggregation in VANETs." In: *Proceedings of the 3rd International Workshop on Vehicular ad hoc Networks - VANET '06*. ACM Press, 2006. ISBN: 1595935401 (cit. on pp. 4, 34, 75, 89, 91, 173).

[146] M. Raya and J.-P. Hubaux. "Securing vehicular ad hoc networks." In: *Journal of Computer Security* 15.1 (Jan. 1, 2007), pp. 39–68 (cit. on pp. 5, 220).

[147] S. S. Al-Riyami and K. G. Paterson. "Certificateless Public Key Cryptography." In: *Advances in Cryptology - ASIACRYPT 2003*. Ed. by C.-S. Laih. Lecture Notes in Computer Science 2894. Springer Berlin Heidelberg, Jan. 1, 2003, pp. 452–473. ISBN: 978-3-540-20592-0, 978-3-540-40061-5 (cit. on pp. 122, 124).

[148] S. Ruj, M. Cavenaghi, Z. Huang, A. Nayak, et al. "On Data-Centric Misbehavior Detection in VANETs." In: *2011 IEEE Vehicular Technology Conference (VTC Fall)*. Sept. 2011, pp. 1–5. DOI: 10.1109/VETECF.2011.6093096 (cit. on p. 6).

[149] H. Saleet, O. Basir, R. Langar, and R. Boutaba. "Region-Based Location-Service-Management Protocol for VANETs." In: *Vehicular Technology, IEEE Transactions on* 59.2 (Feb. 2010), pp. 917–931. ISSN: 0018-9545 (cit. on p. 34).

[150] Y. Sang, H. Shen, Y. Inoguchi, Y. Tan, et al. "Secure Data Aggregation in Wireless Sensor Networks: A Survey." In: *Seventh International Conference on Parallel and Distributed Computing, Applications and Technologies, 2006. PDCAT '06*. Dec. 2006, pp. 315–320. DOI: 10.1109/PDCAT.2006.96 (cit. on pp. 83, 84).

[151] R. A. Santos, R. Edwards, and N. Seed. "Using the cluster-based location routing (CBLR) algorithm for exchanging information on a motorway." In: *4th International Workshop on Mobile and Wireless Communications Network, 2002*. 2002, pp. 212–216. DOI: 10.1109/MWCN.2002.1045724 (cit. on p. 151).

[152] B. Scheuermann, C. Lochert, J. Rybicki, and M. Mauve. "A Fundamental Scalability Criterion for Data Aggregation in VANETs." In: *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking - Mobicom '09*. ACM Press, 2009, p. 285 (cit. on pp. 6, 17, 18, 25, 29, 47, 53, 80, 100, 180, 219).

[153] E. Schoch, F. Kargl, M. Weber, and T. Leinmuller. "Communication patterns in VANETs." In: *Communications Magazine, IEEE* 46.11 (2008), pp. 119–125. ISSN: 0163-6804 (cit. on pp. 13, 15).

[154] E. Schoch and F. Kargl. "On the Efficiency of Secure Beaconing in VANETs." In: *Proceedings of the Third ACM Conference on Wireless Network Security*. WiSec '10. New York, NY, USA: ACM, 2010, pp. 111–116. ISBN: 978-1-60558-923-7. DOI: 10.1145/1741866.1741885 (cit. on pp. 18, 59).

[155] C. Schurgers and M. Srivastava. "Energy efficient routing in wireless sensor networks." In: *IEEE Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force*. Vol. 1. 2001, 357–361 vol.1. DOI: 10.1109/MILCOM.2001.985819 (cit. on pp. 14, 28, 84).

[156] R. S. Schwartz, M. v. Eenennaam, G. Karagiannis, G. Heijenk, et al. "Using V2V Communication to Create over-the-Horizon Awareness in Multiple-Lane Highway Scenarios." In: *Ieee Intelligent Vehicles Symposium (iv) 2010, la Jolla, ca, usa*. la Jolla, ca, usa: IEEE Computer Society Press, June 2010, pp. 998–1005 (cit. on p. 31).

[157] K. Shafiee and V. C. M. Leung. "A Novel Localized Data Aggregation Algorithm for Advanced Vehicular Traffic Information Systems." In: *Communications Workshops, 2009. icc Workshops 2009. Ieee International Conference on*. IEEE, 2009, pp. 1–5 (cit. on p. 34).

[158] R. Shah and J. Rabaey. "Energy aware routing for low energy ad hoc sensor networks." In: *2002 IEEE Wireless Communications and Networking Conference, 2002. WCNC2002*. Vol. 1. Mar. 2002, 350–355 vol.1. DOI: 10.1109/WCNC.2002.993520 (cit. on p. 28).

[159] B. Sklar. "Rayleigh fading channels in mobile digital communication systems .I. Characterization." In: *IEEE Communications Magazine* 35.7 (July 1997), pp. 90–100. ISSN: 0163-6804. DOI: 10.1109/35.601747 (cit. on p. 97).

[160] I. Skog and P. Handel. "In-Car Positioning and Navigation Technologies: A Survey." In: *IEEE Transactions on Intelligent Transportation Systems* 10.1 (Mar. 2009), pp. 4–21. ISSN: 1524-9050. DOI: 10.1109/TITS.2008.2011712 (cit. on p. 12).

[161] L. Stibor, Y. Zang, and H.-J. Reumerman. "Evaluation of Communication Distance of Broadcast Messages in a Vehicular Ad-Hoc Network Using IEEE 802.11p." In: *IEEE Wireless Communications and Networking Conference, 2007.WCNC 2007*. Mar. 2007, pp. 254–257. DOI: 10.1109/WCNC.2007.53 (cit. on p. 14).

[162] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, et al. "An Analysis of a Large Scale Habitat Monitoring Application." In: *Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems*. SenSys '04. New York, NY, USA: ACM, 2004, pp. 214–226. ISBN: 1-58113-879-2. DOI: 10.1145/1031495.1031521 (cit. on p. 83).

[163] M. Takai, J. Martin, and R. Bagrodia. "Effects of Wireless Physical Layer Modeling in Mobile Ad Hoc Networks." In: *Proceedings of the 2Nd ACM International Symposium on Mobile Ad Hoc Networking &Amp; Computing*. MobiHoc '01. New York, NY, USA: ACM, 2001, pp. 87–94. ISBN: 1-58113-428-2. DOI: 10.1145/501426.501429 (cit. on p. 97).

[164] A. Viejo, F. Sebe, and J. Domingo-Ferrer. "Aggregation of Trustworthy Announcement Messages in Vehicular Ad Hoc Networks." In: *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*. Apr. 2009, pp. 1–5. DOI: 10.1109/VETECS.2009.5073371 (cit. on p. 86).

[165] X. Wang and P. Tague. "ASIA: Accelerated secure in-network aggregation in vehicular sensing networks." In: *2013 10th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*. June 2013, pp. 514–522. DOI: 10.1109/SAHCN.2013.6645023 (cit. on pp. 90, 91).

[166] T. Welch. "A Technique for High-Performance Data Compression." In: *IEEE Computer* 17.6 (1984), pp. 8–19. ISSN: 0018-9162. DOI: 10.1109/MC.1984.1659158 (cit. on p. 32).

[167] K.-Y. Whang, B. T. Vander-Zanden, and H. M. Taylor. "A Linear-time Probabilistic Counting Algorithm for Database Applications." In: *ACM Trans. Database Syst.* 15.2 (June 1990), pp. 208–229. ISSN: 0362-5915. DOI: 10.1145/78922.78925 (cit. on p. 133).

[168] M. G. Wing, A. Eklund, and L. D. Kellogg. "Consumer-Grade Global Positioning System (GPS) Accuracy and Reliability." In: *Journal of Forestry* 103.4 (June 1, 2005), pp. 169–173 (cit. on p. 26).

[169] L. Wischhof, A. Ebner, and H. Rohling. "Information Dissemination in Self-Organizing Intervehicle Networks." In: *IEEE Transactions on Intelligent Transportation Systems* 6.1 (Mar. 2005), pp. 90–101 (cit. on pp. 26, 29, 40, 47, 48).

[170] L. Wischhof, A. Ebner, H. Rohling, M. Lott, et al. "SOTIS - a Self Organizing Traffic Information System." In: *the 57th Ieee Semiannual Vehicular Technology Conference, 2003. vtc 2003-Spring*. Ieee, 2003, pp. 2442–2446. ISBN: 0-7803-7757-5 (cit. on pp. 25, 29, 40, 47, 48, 150).

[171] M. Wolf and T. Gendrullis. "Design, Implementation, and Evaluation of a Vehicular Hardware Security Module." In: *Information Security and Cryptology - ICISC 2011*. Ed. by H. Kim. Lecture Notes in Computer Science 7259. Springer Berlin Heidelberg, Jan. 1, 2012, pp. 302–318. ISBN: 978-3-642-31911-2, 978-3-642-31912-9 (cit. on pp. 14, 67, 79).

[172] C. Wu, Y. Xing, X. Chen, D. Long, et al. "Generic On-line/Off-line Aggregate Signatures." In: *International Conference on Embedded Software and Systems Symposia, 2008. ICESS Symposia '08*. July 2008, pp. 107–112. DOI: 10.1109/ICESS.Symposia.2008.56 (cit. on p. 73).

[173] B. Xu, A. M. Ouksel, and O. Wolfson. "Opportunistic Resource Exchange in Inter-Vehicle Ad-Hoc Networks." In: *Mobile Data Management*. 2004, pp. 4–12 (cit. on p. 34).

[174] N. Xu. "A Survey of Sensor Network Applications." In: *IEEE Communications Magazine* 40 (2002) (cit. on pp. 28, 84).

[175] T. Yamashita, K. Izumi, K. Kurumatani, and H. Nakashima. "Smooth Traffic Flow with a Cooperative Car Navigation System." In: *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*. AAMAS '05. New York, NY, USA: ACM, 2005, pp. 478–485. ISBN: 1-59593-093-0. DOI: 10.1145/1082473.1082546 (cit. on p. 13).

[176] B. Yu, J. Gong, and C.-Z. Xu. "Catch-up: a Data Aggregation Scheme for VANETs." In: *Proceedings of the Fifth acm International Workshop on Vehicular Inter-Networking*. ACM, 2008, pp. 49–57 (cit. on pp. 35, 41).

[177] B. Yu, C.-Z. Xu, and M. Guo. "Adaptive Forwarding Delay Control for VANET Data Aggregation." In: *Parallel and Distributed Systems, IEEE Transactions on* 23.1 (2012), pp. 11–18 (cit. on pp. 35, 41).

[178] A. Yun, J. H. Cheon, and Y. Kim. "On Homomorphic Signatures for Network Coding." In: *IEEE Transactions on Computers* 59.9 (Sept. 2010), pp. 1295–1296. ISSN: 0018-9340. DOI: 10.1109/TC.2010.73 (cit. on p. 75).

[179] D. Zekri, B. Defude, and T. Delot. "A Cooperative Scheme to Aggregate Spatio-Temporal Events in VANETs." In: *Proceedings of the 16th International Database Engineering & Applications Sysmposium*. 2012, pp. 100–109 (cit. on p. 33).

[180] X. Zeng, R. Bagrodia, and M. Gerla. "GloMoSim: a library for parallel simulation of large-scale wireless networks." In: *Twelfth Workshop on Parallel and Distributed Simulation, 1998. PADS 98. Proceedings*. May 1998, pp. 154–161. DOI: 10.1109/PADS.1998.685281 (cit. on p. 97).

[181] H. Zhu, X. Lin, R. Lu, P.-H. Ho, et al. "AEMA: An Aggregated Emergency Message Authentication Scheme for Enhancing the Security of Vehicular Ad Hoc Networks." In: *IEEE International Conference on Communications, 2008. ICC '08*. May 2008, pp. 1436–1440. DOI: 10.1109/ICC.2008.278 (cit. on p. 23).

[182] H. Zhu, X. Lin, R. Lu, P.-H. Ho, et al. "AEMA: An Aggregated Emergency Message Authentication Scheme for Enhancing the Security of Vehicular Ad Hoc Networks." In: *IEEE International Conference on Communications, 2008. ICC '08*. May 2008, pp. 1436–1440. DOI: 10.1109/ICC.2008.278 (cit. on p. 86).

[183] S. Zionts and J. Wallenius. "An Interactive Programming Method for Solving the Multiple Criteria Problem." In: *Management Science* 22.6 (1976) (cit. on p. 31).

[184] J. Ziv and A. Lempel. "A Universal Algorithm for Sequential Data Compression." In: *Information Theory, IEEE Transactions on* 23.3 (May 1977), pp. 337–343. ISSN: 0018-9448 (cit. on p. 26).

OTHER REFERENCES

[185] R. Barr. "An Efficient, Unifying Approach to Simulation Using Virtual Machnies." PhD thesis. Cornell University, 2004 (cit. on p. 96).

[186] *Bundesautobahn 7*. In: *Wikipedia*. Page Version ID: 127203064. Mar. 3, 2014. URL: http://de.wikipedia.org/w/index.php?title=Bundesautobahn_7%5C&oldid=127203064 (visited on 03/12/2014) (cit. on pp. 13, 100).

[187] S. Dietzel. "Fehlertolerante Sicherheitsmechanismen für Aggregation in Fahrzeug-Fahrzeug-Netzen." Diplom thesis. University of Ulm, 2008 (cit. on p. 109).

[188] S. Dietzel. "Privacy Implications of in-Network Aggregation Mechanisms for VANETs (Invited Paper)." In: *8th International Conference on Wireless on-Demand Network Systems and Services (WONS)*. 2011, pp. 91–95. DOI: 10.1109/WONS.2011.5720205 (cit. on pp. 8, 61).

[189] "Digital signatures including identity-based aggregate signatures." C. B. Gentry and Z. A. Ramzan. U.S. Classification 713/176, 380/277, 380/28, 380/59. Feb. 16, 2010 (cit. on pp. 73, 74, 120, 122, 123).

[190] ETSI. *Intelligent Transport Systems (ITS); Communications Architecture*. EN 302 665. 2010 (cit. on p. 67).

[191] ETSI. *Intelligent Transport Systems (ITS); Radiocommunications equipment operating in the 5 855 MHz to 5 925 MHz frequency band; Harmonized EN covering the essential requirements of article 3.2 of the R&TTE Directive*. EN 302 571. 2013 (cit. on p. 13).

[192] ETSI. *Intelligent Transport Systems (ITS); Security; ITS communications security architecture and security management*. TS 102 940. 2012 (cit. on pp. 5, 57, 60, 67, 77, 79).

[193] ETSI. *Intelligent Transport Systems (ITS); Security; Security header and certificate formats*. TS 103 097. 2013 (cit. on pp. 57, 59, 79).

[194] ETSI. *Intelligent Transport Systems (ITS); Security; Threat, Vulnerability and Risk Analysis (TVRA)*. TR 102 893. 2010 (cit. on p. 57).

[195] ETSI. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic set of Applications*. TS 102 637. 2010 (cit. on p. 151).

[196] ETSI. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Local Dynamic Map (LDM); Rationale for and guidance on standardization*. TR 102 863. 2011 (cit. on p. 15).

[197] ETSI. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 1: Functional Requirements*. TS 102 637-1. 2010 (cit. on p. 4).

[198] ETSI. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 1: Functional Requirements*. EN 302 637-1. 2010 (cit. on p. 12).

[199] ETSI. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service*. EN 302 637-2. 2013 (cit. on pp. 15, 17).

[200] ETSI. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service*. TS 102 637-3. 2010 (cit. on p. 53).

[201] ETSI. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service*. EN 302 637-3. 2013 (cit. on pp. 17, 53).

[202] ETSI. *Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media Independent Functionality*. EN 302 636-4-1. 2013 (cit. on pp. 4, 16).

[203] ETSI. *Intelligent Transport Systems; OSI cross-layer topics; Part 8: Interface between security entity and network and transport layers*. TS 102 723-8. (to be published) (cit. on p. 67).

[204] *European Motor Vehicle Parc*. ANFAC, 2011. URL: http://www.acea.be/uploads/statistic_documents/2013_ANFAC_Report.pdf (visited on 11/06/2014) (cit. on p. 3).

[205] *Framework for the deployment of Intelligent Transport Systems in the field of road transport and for interfaces with other modes of transport*. Directive 2010/40/EU. European Parliament, 2010. URL: http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32010L0040 (cit. on p. 3).

[206] *Free Community-based Mapping, Traffic & Navigation App*. URL: https://www.waze.com/ (visited on 11/11/2014) (cit. on p. 12).

[207] C. Gentry. "A fully homomorphic encryption scheme." PhD thesis. Stanford University, 2009 (cit. on pp. 75, 85).

[208] A. Goldsmith. *Wireless communications*. Cambridge; New York: Cambridge University Press, 2005. ISBN: 1601197640 (cit. on p. 97).

[209] R. W. van der Heijden. "SeDyA: secure dynamic aggregation in VANETs." Master thesis. University of Twente, Aug. 10, 2012 (cit. on p. 131).

[210] *IEEE Standard for Information technology – Telecommunications and information exchange between systems Local and metropolitan area networks – Specific requirements; Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. IEEE Std. 802.11-2012. 2012 (cit. on pp. 13, 18, 73, 97, 101).

[211] T. Imielinski and J. Navas. *GPS-Based Addressing and Routing*. RFC 2009. 1996. URL: http://tools.ietf.org/html/rfc2009 (visited on 02/11/2014) (cit. on p. 15).

[212] *Intelligent Transportation Systems (ITS) Standards Program Strategic Plan for 2011–2014*. FHWA-JPO-11-052. U.S. Department of Transportation, 2011. URL: http://www.its.dot.gov/standards_strategic_plan/ (cit. on p. 3).

[213] ISO. *Traffic and Traveller Information (TTI) – TTI messages via traffic message coding – Part 1: Coding protocol for Radio Data System – Traffic Message Channel (RDS-TMC) using ALERT-C*. ISO 14819-1:2003 (cit. on p. 12).

[214] ISO/IEC. *Coding of audio-visual objects*. ISO/IEC 14496-2. 2009 (cit. on p. 22).

[215] ISO/IEC. *Digital compression and coding of continuous-tone still images: Requirements and guidelines*. ISO/IEC 10918-1. 1994 (cit. on p. 22).

[216] ISO/IEC. *Generic coding of moving pictures and associated audio information – Part 3: Audio*. ISO/IEC 13818-3. 1998 (cit. on pp. 22, 32).

[217] ISO/IEC. *Information technology – Database languages – SQL*. ISO/IEC 9075:2011. 2011 (cit. on p. 27).

[218] ISO/IEC. *Intelligent transport systems - Communications access for land mobiles (CALM) - Architecture*. ISO 21217:2014. 2014 (cit. on p. 15).

[219] A. Jøsang. *Subjective Logic*. University of Oslo, Feb. 2014. URL: http://folk.uio.no/josang/papers/subjective_logic.pdf (cit. on pp. 193, 195, 206).

[220] Julian Gürtler. "Sicherung der Datenintegrität bei Nachrichtenaggregation in VANETs." Master thesis. University of Ulm, Nov. 29, 2013 (cit. on p. 169).

[221] W. Klein Wolterink. "Location-based Forwarding in Vehicular Networks." PhD thesis. University of Twente, 2013 (cit. on p. 4).

[222] H. Krawczyk, R. Canetti, and M. Bellare. *HMAC: Keyed-Hashing for Message Authentication*. RFC 2104. IETF, 1997. URL: http://tools.ietf.org/html/rfc2104 (visited on 02/12/2014) (cit. on p. 85).

[223] C. Lochert, B. Scheuermann, and M. Mauve. "Information Dissemination in VANETs." In: H. Hartenstein and K. Laberteaux. *VANET Vehicular Applications and Inter-Networking Technologies*. John Wiley & Sons, Nov. 4, 2009. ISBN: 9780470740620 (cit. on pp. 4, 14).

[224] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. 1 edition. New York: Cambridge University Press, July 7, 2008. 496 pp. ISBN: 9780521865715 (cit. on p. 174).

[225] "Method of providing digital signatures." US4309569 A. R. C. Merkle. U.S. Classification 713/177, 713/180, 902/2, 340/5.8, 902/5. Jan. 5, 1982 (cit. on pp. 76, 84, 88).

[226] Mihail Balanici. "Cluster-based Aggregation for Inter-vehicle Communication." Master thesis. University of Ulm, 2013 (cit. on p. 149).

[227] *Public Key Infrastructure*. Memo. C2C-CC, May 2012. URL: http://www.car-to-car.org/ (cit. on p. 57).

[228] D. Salomon. *Data Compression: the Complete Reference*. 10. Springer, 2007. ISBN: 9781846286025 (cit. on p. 31).

[229] J. H. Schiller. *Mobile communications*. London: Addison-Wesley, 2011. ISBN: 0321123816 9780321123817 (cit. on p. 13).

[230] E. Schoch. *Secure Communication in Inter-Vehicle Networks*. München: Dr. Hut, 2009. 257 pp. ISBN: 9783868532715 (cit. on p. 96).

[231] P. Scholl. *Autobahnatlas Online*. URL: http://www.autobahnatlas-online.de/ (visited on 03/12/2014) (cit. on pp. 13, 100).

[232] R. Shirey. *Internet Security Glossary Version 2*. RFC 4949. IETF, 2007. URL: https://tools.ietf.org/html/rfc4949 (cit. on pp. 5, 59, 60).

[233] *Specification for Telecommunications and Information Exchange Between Roadside and Vehicle Systems – 5 GHz Band Dedicated Short Range Communications (DSRC) Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. ASTM International, 2010. URL: http://www.astm.org/Standards/E2213.htm (visited on 11/11/2014) (cit. on p. 15).

[234] *Standardisation Mandate Addressed to cen, Cenelec and Etsi In the Field of Information and Communication Technologies to Support the Interoperability of Co-operative Systems for Intelligent Transport in the European Community*. Mandate M/453. European Commission, Oct. 6, 2009. URL: http://ec.europa.eu/enterprise/sectors/ict/files/standardisation_mandate_en.pdf (cit. on p. 12).

[235] *The ENVIBASE-Project: Documentation / Online Handbook*. URL: http://www.stadtentwicklung.berlin.de/archiv_sensut/umwelt/uisonline/envibase/handbook/ (visited on 11/11/2014) (cit. on p. 13).

[236] *TomTom Live Services: TomTom Traffic*. URL: http://www.tomtom.com/en_gb/services/live/hd-traffic/ (visited on 11/11/2014) (cit. on p. 12).

[237] *Towards a European road safety area: policy orientations on road safety 2011-2020*. COM(2010) 389. European Commission, 2010. URL: http://ec.europa.eu/transport/road_safety/pdf/road_safety_citizen/road_safety_citizen_100924_en.pdf (cit. on p. 3).

[238] USDOT. *Vehicle safety communications project: final report*. DOT HS 810 591. 2006 (cit. on p. 11).

[239] A. Vahdat and D. Becker. *Epidemic Routing for Partially-Connected ad hoc Networks*. Duke University (cit. on p. 34).

[240] W. m. Waggener. *Pulse Code Modulation Techniques: With Applications in Communications and Data Recording*. van Nostrand Reinhold, 1995. ISBN: 9780442014360 (cit. on p. 31).

[241] *Wat is de maximumsnelheid voor het wegverkeer?* URL: http://www.rijksoverheid.nl/onderwerpen/wegen/vraag-en-antwoord/wat-is-de-maximumsnelheid-voor-het-wegverkeer.html (visited on 12/14/2014) (cit. on p. 28).

[242] L. A. Zadeh, G. J. Klir, and B. Yuang. *Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems*. World Scientific, 1996 (cit. on pp. 50, 51, 115).

[243] *.ZIP File Format Specification*. PKWARE Inc., APPNOTE.TXT version 6.3.3. 2012 (cit. on p. 32).

M.H.G. Verhoef. *Modeling and Validating Distributed Embedded Real-Time Control Systems*. Faculty of Science, Mathematics and Computer Science, RU. 2009-01

M. de Mol. *Reasoning about Functional Programs: Sparkle, a proof assistant for Clean*. Faculty of Science, Mathematics and Computer Science, RU. 2009-02

M. Lormans. *Managing Requirements Evolution*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2009-03

M.P.W.J. van Osch. *Automated Model-based Testing of Hybrid Systems*. Faculty of Mathematics and Computer Science, TU/e. 2009-04

H. Sozer. *Architecting Fault-Tolerant Software Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2009-05

M.J. van Weerdenburg. *Efficient Rewriting Techniques*. Faculty of Mathematics and Computer Science, TU/e. 2009-06

H.H. Hansen. *Coalgebraic Modelling: Applications in Automata Theory and Modal Logic*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2009-07

A. Mesbah. *Analysis and Testing of Ajax-based Single-page Web Applications*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2009-08

A.L. Rodriguez Yakushev. *Towards Getting Generic Programming Ready for Prime Time*. Faculty of Science, UU. 2009-9

K.R. Olmos Joffré. *Strategies for Context Sensitive Program Transformation*. Faculty of Science, UU. 2009-10

J.A.G.M. van den Berg. *Reasoning about Java programs in PVS using JML*. Faculty of Science, Mathematics and Computer Science, RU. 2009-11

M.G. Khatib. *MEMS-Based Storage Devices. Integration in Energy-Constrained Mobile Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2009-12

S.G.M. Cornelissen. *Evaluating Dynamic Analysis Techniques for Program Comprehension*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2009-13

D. Bolzoni. *Revisiting Anomaly-based Network Intrusion Detection Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2009-14

H.L. Jonker. *Security Matters: Privacy in Voting and Fairness in Digital Exchange*. Faculty of Mathematics and Computer Science, TU/e. 2009-15

M.R. Czenko. *TuLiP - Reshaping Trust Management*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2009-16

T. Chen. *Clocks, Dice and Processes*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2009-17

C. Kaliszyk. *Correctness and Availability: Building Computer Algebra on top of Proof Assistants and making Proof Assistants available over the Web*. Faculty of Science, Mathematics and Computer Science, RU. 2009-18

R.S.S. O'Connor. *Incompleteness & Completeness: Formalizing Logic and Analysis in Type Theory*. Faculty of Science, Mathematics and Computer Science, RU. 2009-19

B. Ploeger. *Improved Verification Methods for Concurrent Systems*. Faculty of Mathematics and Computer Science, TU/e. 2009-20

T. Han. *Diagnosis, Synthesis and Analysis of Probabilistic Models*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2009-21

R. Li. *Mixed-Integer Evolution Strategies for Parameter Optimization and Their Applications to Medical Image Analysis*. Faculty of Mathematics and Natural Sciences, UL. 2009-22

J.H.P. Kwisthout. *The Computational Complexity of Probabilistic Networks*. Faculty of Science, UU. 2009-23

T.K. Cocx. *Algorithmic Tools for Data-Oriented Law Enforcement*. Faculty of Mathematics and Natural Sciences, UL. 2009-24

A.I. Baars. *Embedded Compilers*. Faculty of Science, UU. 2009-25

M.A.C. Dekker. *Flexible Access Control for Dynamic Collaborative Environments*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2009-26

J.F.J. Laros. *Metrics and Visualisation for Crime Analysis and Genomics*. Faculty of Mathematics and Natural Sciences, UL. 2009-27

C.J. Boogerd. *Focusing Automatic Code Inspections*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2010-01

M.R. Neuhäußer. *Model Checking Nondeterministic and Randomly Timed Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2010-02

J. Endrullis. *Termination and Productivity*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2010-03

T. Staijen. *Graph-Based Specification and Verification for Aspect-Oriented Languages*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2010-04

Y. Wang. *Epistemic Modelling and Protocol Dynamics*. Faculty of Science, UvA. 2010-05

J.K. Berendsen. *Abstraction, Prices and Probability in Model Checking Timed Automata*. Faculty of Science, Mathematics and Computer Science, RU. 2010-06

A. Nugroho. *The Effects of UML Modeling on the Quality of Software*. Faculty of Mathematics and Natural Sciences, UL. 2010-07

A. Silva. *Kleene Coalgebra*. Faculty of Science, Mathematics and Computer Science, RU. 2010-08

J.S. de Bruin. *Service-Oriented Discovery of Knowledge - Foundations, Implementations and Applications*. Faculty of Mathematics and Natural Sciences, UL. 2010-09

D. Costa. *Formal Models for Component Connectors*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2010-10

M.M. Jaghoori. *Time at Your Service: Schedulability Analysis of Real-Time and Distributed Services*. Faculty of Mathematics and Natural Sciences, UL. 2010-11

R. Bakhshi. *Gossiping Models: Formal Analysis of Epidemic Protocols*. Faculty of Sciences, Department of Computer Science, VUA. 2011-01

B.J. Arnoldus. *An Illumination of the Template Enigma: Software Code Generation with Templates*. Faculty of Mathematics and Computer Science, TU/e. 2011-02

E. Zambon. *Towards Optimal IT Availability Planning: Methods and Tools*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2011-03

L. Astefanoaei. *An Executable Theory of Multi-Agent Systems Refinement*. Faculty of Mathematics and Natural Sciences, UL. 2011-04

J. Proença. *Synchronous coordination of distributed components*. Faculty of Mathematics and Natural Sciences, UL. 2011-05

A. Moralı. *IT Architecture-Based Confidentiality Risk Assessment in Networks of Organizations*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2011-06

M. van der Bijl. *On changing models in Model-Based Testing*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2011-07

C. Krause. *Reconfigurable Component Connectors*. Faculty of Mathematics and Natural Sciences, UL. 2011-08

M.E. Andrés. *Quantitative Analysis of Information Leakage in Probabilistic and Nondeterministic Systems*. Faculty of Science, Mathematics and Computer Science, RU. 2011-09

M. Atif. *Formal Modeling and Verification of Distributed Failure Detectors*. Faculty of Mathematics and Computer Science, TU/e. 2011-10

P.J.A. van Tilburg. *From Computability to Executability – A process-theoretic view on automata theory*. Faculty of Mathematics and Computer Science, TU/e. 2011-11

Z. Protic. *Configuration management for models: Generic methods for model comparison and model co-evolution*. Faculty of Mathematics and Computer Science, TU/e. 2011-12

S. Georgievska. *Probability and Hiding in Concurrent Processes*. Faculty of Mathematics and Computer Science, TU/e. 2011-13

S. Malakuti. *Event Composition Model: Achieving Naturalness in Runtime Enforcement*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2011-14

M. Raffelsieper. *Cell Libraries and Verification*. Faculty of Mathematics and Computer Science, TU/e. 2011-15

C.P. Tsirogiannis. *Analysis of Flow and Visibility on Triangulated Terrains*. Faculty of Mathematics and Computer Science, TU/e. 2011-16

Y.-J. Moon. *Stochastic Models for Quality of Service of Component Connectors*. Faculty of Mathematics and Natural Sciences, UL. 2011-17

R. Middelkoop. *Capturing and Exploiting Abstract Views of States in OO Verification*. Faculty of Mathematics and Computer Science, TU/e. 2011-18

M.F. van Amstel. *Assessing and Improving the Quality of Model Transformations*. Faculty of Mathematics and Computer Science, TU/e. 2011-19

A.N. Tamalet. *Towards Correct Programs in Practice*. Faculty of Science, Mathematics and Computer Science, RU. 2011-20

H.J.S. Basten. *Ambiguity Detection for Programming Language Grammars*. Faculty of Science, UvA. 2011-21

M. Izadi. *Model Checking of Component Connectors*. Faculty of Mathematics and Natural Sciences, UL. 2011-22

L.C.L. Kats. *Building Blocks for Language Workbenches*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2011-23

S. Kemper. *Modelling and Analysis of Real-Time Coordination Patterns*. Faculty of Mathematics and Natural Sciences, UL. 2011-24

J. Wang. *Spiking Neural P Systems*. Faculty of Mathematics and Natural Sciences, UL. 2011-25

A. Khosravi. *Optimal Geometric Data Structures*. Faculty of Mathematics and Computer Science, TU/e. 2012-01

A. Middelkoop. *Inference of Program Properties with Attribute Grammars, Revisited*. Faculty of Science, UU. 2012-02

Z. Hemel. *Methods and Techniques for the Design and Implementation of Domain-Specific Languages*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2012-03

T. Dimkov. *Alignment of Organizational Security Policies: Theory and Practice*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2012-04

S. Sedghi. *Towards Provably Secure Efficiently Searchable Encryption*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2012-05

F. Heidarian Dehkordi. *Studies on Verification of Wireless Sensor Networks and Abstraction Learning for System Inference*. Faculty of Science, Mathematics and Computer Science, RU. 2012-06

K. Verbeek. *Algorithms for Cartographic Visualization*. Faculty of Mathematics and Computer Science, TU/e. 2012-07

D.E. Nadales Agut. *A Compositional Interchange Format for Hybrid Systems: Design and Implementation*. Faculty of Mechanical Engineering, TU/e. 2012-08

H. Rahmani. *Analysis of Protein-Protein Interaction Networks by Means of Annotated Graph Mining Algorithms*. Faculty of Mathematics and Natural Sciences, UL. 2012-09

S.D. Vermolen. *Software Language Evolution*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2012-10

L.J.P. Engelen. *From Napkin Sketches to Reliable Software*. Faculty of Mathematics and Computer Science, TU/e. 2012-11

F.P.M. Stappers. *Bridging Formal Models – An Engineering Perspective*. Faculty of Mathematics and Computer Science, TU/e. 2012-12

W. Heijstek. *Software Architecture Design in Global and Model-Centric Software Development*. Faculty of Mathematics and Natural Sciences, UL. 2012-13

C. Kop. *Higher Order Termination*. Faculty of Sciences, Department of Computer Science, VUA. 2012-14

A. Osaiweran. *Formal Development of Control Software in the Medical Systems Domain*. Faculty of Mathematics and Computer Science, TU/e. 2012-15

W. Kuijper. *Compositional Synthesis of Safety Controllers*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2012-16

H. Beohar. *Refinement of Communication and States in Models of Embedded Systems*. Faculty of Mathematics and Computer Science, TU/e. 2013-01

G. Igna. *Performance Analysis of Real-Time Task Systems using Timed Automata*. Faculty of Science, Mathematics and Computer Science, RU. 2013-02

E. Zambon. *Abstract Graph Transformation – Theory and Practice*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2013-03

B. Lijnse. *TOP to the Rescue – Task-Oriented Programming for Incident Response Applications*. Faculty of Science, Mathematics and Computer Science, RU. 2013-04

G.T. de Koning Gans. *Outsmarting Smart Cards*. Faculty of Science, Mathematics and Computer Science, RU. 2013-05

M.S. Greiler. *Test Suite Comprehension for Modular and Dynamic Systems*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2013-06

L.E. Mamane. *Interactive mathematical documents: creation and presentation*. Faculty of Science, Mathematics and Computer Science, RU. 2013-07

M.M.H.P. van den Heuvel. *Composition and synchronization of real-time components upon one processor*. Faculty of Mathematics and Computer Science, TU/e. 2013-08

J. Businge. *Co-evolution of the Eclipse Framework and its Third-party Plug-ins*. Faculty of Mathematics and Computer Science, TU/e. 2013-09

S. van der Burg. *A Reference Architecture for Distributed Software Deployment*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2013-10

J.J.A. Keiren. *Advanced Reduction Techniques for Model Checking*. Faculty of Mathematics and Computer Science, TU/e. 2013-11

D.H.P. Gerrits. *Pushing and Pulling: Computing push plans for disk-shaped robots, and dynamic labelings for moving points*. Faculty of Mathematics and Computer Science, TU/e. 2013-12

M. Timmer. *Efficient Modelling, Generation and Analysis of Markov Automata*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2013-13

M.J.M. Roeloffzen. *Kinetic Data Structures in the Black-Box Model*. Faculty of Mathematics and Computer Science, TU/e. 2013-14

L. Lensink. *Applying Formal Methods in Software Development*. Faculty of Science, Mathematics and Computer Science, RU. 2013-15

C. Tankink. *Documentation and Formal Mathematics — Web Technology meets Proof Assistants*. Faculty of Science, Mathematics and Computer Science, RU. 2013-16

C. de Gouw. *Combining Monitoring with Runtime Assertion Checking*. Faculty of Mathematics and Natural Sciences, UL. 2013-17

J. van den Bos. *Gathering Evidence: Model-Driven Software Engineering in Automated Digital Forensics*. Faculty of Science, UvA. 2014-01

D. Hadziosmanovic. *The Process Matters: Cyber Security in Industrial Control Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2014-02

A.J.P. Jeckmans. *Cryptographically-Enhanced Privacy for Recommender Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2014-03

C.-P. Bezemer. *Performance Optimization of Multi-Tenant Software Systems*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2014-04

T.M. Ngo. *Qualitative and Quantitative Information Flow Analysis for Multi-threaded Programs*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2014-05

A.W. Laarman. *Scalable Multi-Core Model Checking*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2014-06

J. Winter. *Coalgebraic Characterizations of Automata-Theoretic Classes*. Faculty of Science, Mathematics and Computer Science, RU. 2014-07

W. Meulemans. *Similarity Measures and Algorithms for Cartographic Schematization*. Faculty of Mathematics and Computer Science, TU/e. 2014-08

A.F.E. Belinfante. *JTorX: Exploring Model-Based Testing*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2014-09

A.P. van der Meer. *Domain Specific Languages and their Type Systems*. Faculty of Mathematics and Computer Science, TU/e. 2014-10

B.N. Vasilescu. *Social Aspects of Collaboration in Online Software Communities*. Faculty of Mathematics and Computer Science, TU/e. 2014-11

F.D. Aarts. *Tomte: Bridging the Gap between Active Learning and Real-World Systems*. Faculty of Science, Mathematics and Computer Science, RU. 2014-12

N. Noroozi. *Improving Input-Output Conformance Testing Theories*. Faculty of Mathematics and Computer Science, TU/e. 2014-13

M. Helvensteijn. *Abstract Delta Modeling: Software Product Lines and Beyond*. Faculty of Mathematics and Natural Sciences, UL. 2014-14

P. Vullers. *Efficient Implementations of Attribute-based Credentials on Smart Cards*. Faculty of Science, Mathematics and Computer Science, RU. 2014-15

F.W. Takes. *Algorithms for Analyzing and Mining Real-World Graphs*. Faculty of Mathematics and Natural Sciences, UL. 2014-16

M.P. Schraagen. *Aspects of Record Linkage*. Faculty of Mathematics and Natural Sciences, UL. 2014-17

G. Alpár. *Attribute-Based Identity Management: Bridging the Cryptographic Design of ABCs with the Real World*. Faculty of Science, Mathematics and Computer Science, RU. 2015-01

A.J. van der Ploeg. *Efficient Abstractions for Visualization and Interaction*. Faculty of Science, UvA. 2015-02

R.J.M. Theunissen. *Supervisory Control in Health Care Systems*. Faculty of Mechanical Engineering, TU/e. 2015-03

T.V. Bui. *A Software Architecture for Body Area Sensor Networks: Flexibility and Trustworthiness*. Faculty of Mathematics and Computer Science, TU/e. 2015-04

A. Guzzi. *Supporting Developers' Teamwork from within the IDE*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2015-05

T. Espinha. *Web Service Growing Pains: Understanding Services and Their Clients*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2015-06

S. Dietzel. *Resilient In-network Aggregation for Vehicular Networks*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2015-07